# INDEX

# Acknowledgment

I have completed my assignment by learning from the following sources

1 The base of the Whole programme is based on these links
   a) https://www.youtube.com/channel/UCFbNIlppjAuEX4znoulh0Cw
   b) https://www.youtube.com/watch?v=rxzOqP9YwmM
   c) https://www.youtube.com/watch?v=UymGJnv-WsE
   d) https://www.youtube.com/watch?v=tBr-PybP_9c
   e) https://www.tutorialspoint.com/socket.io/socket.io_chat_application.htm
2) https://www.tutorialspoint.com/socket.io/index.htm
3) https://www.w3schools.com/nodejs/
4) https://www.tutorialspoint.com/nodejs/index.htm
5) https://www.w3schools.com/js/DEFAULT.asp
6) https://www.tutorialspoint.com/javascript/index.htm
7) https://www.w3schools.com/html/
8) https://www.tutorialspoint.com/html/index.htm
9) https://www.w3schools.com/REACT/DEFAULT.ASP
10) https://www.tutorialspoint.com/reactjs/index.htm
11) https://www.geeksforgeeks.org/
12) https://www.tutorialspoint.com/expressjs/index.htm
13) Other Books and study materials
14) Given Study Materials
15) Previous Assignments

NOTE – I have not copied anything from these sites, such as programs. Only have taken the idea of the general syntax and working of the programme. Also, this is my first time in web development, so I am in the learning phase, so some code may match (not exactly).

# Pre-Requisite

## How To Make Executable and Run Programme

1) Unzip File at location x
2) Open Terminal and change directory to that location x
3) Now enter the command.
   export PATH=$PATH:/home/badaalvm/node -v x.x.x-Linux-64/bin
4) Now enter npm run startchat
   It will run nodemon and nodejs
5) Some login Ids are
   a) Mohit -asdf
   b) Cohit -asdf
   c) Dohit -asdf

## Structure Of Programme

Program is made up of directory structure which includes

1) views – Source Folder for all EJS (HTML file).
   a) Frame.ejs – contains Login page at "/"
   b) Grpselect.ejs – contains group selection and broadcast page at '/grselect'
   c) Grssee.ejs - contains user and group page at '/:GropNumber'
2) Node Modules – contains all local modules I tried to implement in my programme
3) Chatapp.js – Main Nodejs and server file at port 5000
4) Package.json file – contains json file
5) .nmprc – contains proxy info to connect npm to net
6) Client – contains react App

# Chat Application

Very useful nowadays is a Chat Application which helps us in sharing ideas, jokes, videos, etc.

There are several types of OS based chat applications like android applications, windows etc. But the chat App should be platform-independent and should be accessed by all in an easy way.

So the easiest way to do this is to make a Web Chat Application.

## Types of Services Used

**Node JS –** It constitutes the body of our programme. Gives functionality of HTTPS. Express Js app development easiness. MongoDB type database. Freedom to acquire different libraries and run server and backend part of our App

**HTML –** It constitutes the visual representation of our App on the client-side with the help of ejs. It makes a basic login page and broadcast and group and individual chat page with the added functionality of link, forum and buttons to get input from users.

**Express JS-** It gives API to develop internet applications. It helps in controlling events and using Javascript easily on both sides. Its add functionality of get and post.

**Socket.io-** It is the soul of my programme. It helps in passing info from client to server and vice versa without refreshing the page with the help of preventdefault.

**Nodemon-** It helps in running the Server again and again.

**MongoDB-** I have tried to implement this to provide the database for chat of individual, login ids and group ids and password.

**JavaScript-** To write all functions and conditions. All Node.Js React is based on this

**HTTP-** To create a server and listen to specific ports

**EJS-** To embed java template and pass data in HTML

## Types of Function/Tags Used

**.Set –** To use Embedded JavaScript Template in HTML and to insert data.

**Express.urlencoded –** A middleware to pass info in post

**Express.get –** Used to send info to the client

**Express.post –** Used to get info to the client

**.render –** Used to render HTML pages

**.redirect –** Used to redirect to seme pages which uses get function

**Console.log –** Used to create a log in terminals

**Socket.on –** Used to listen to sockets events and execute function accordingly

**Socket.emit –** Used to send action and data

**Socket.emit –** Used to send action and data

**HTTP.listen –** Used to display port on which server will listen to

**event.preventdefault –** To prevent default actions like page reloading on post.

**.createElement–** To create tags element and write content in with the help of .innerhtml, .innertext.

**.getElementById–** To access a particular tag and its content and append some material in it using .append.

**<a href>–** To access a particular tag and its content and append some material in it using .append.

**<Img>/<Video>/<Audio>–** To access media on web pages.

**Java Script File Reader–** To access a particular file and read it and save it as a URL or text on client-side

**File Reader.onload–** To take appropriate action on a load of file

**<form>–** To create a form with buttons action and post method and input fields

# Result

1) I have made a Web that performs all significant functions of chatting, like chatting with individuals and in groups.
2) It also helps in sending media of almost all types like image reels gif mp4 etc. Audio features can b added with some extra-same code as video.
3) Users can also create new groups with passwords, and the functionality of accessing groups with the password can be made available with a prompt with some extra codes.
4) Users can login easily with their id and password. Also, we can add extra functionality of adding a user with copying same code as a group add in frame.ejs or creating new HTML
5) Users can send a broadcast message when they login but can also use the link to join a group or click on names to chat with individuals
6) Note logout etc. features are not embedded, and the back button of the browser is to be not used
7) Security can be enhanced by using render instead of redirect in my programme.
8) Images are saved as URL in my programme. Similarly, Video and other media are also saved as Url in my programme
9) I have implemented MongoDB, but due to some VM constraints, it is not working properly sometimes.
10) I have the concept of local storage and global storage, which works and stores chat data and other data of group and user.
11) I am not able to plot graphs for requests as unable to handle time properly in my programme.
12) Data remain if a user logout and become disconnected and show up in the same sequence when the user login again.
13) I have used only simple HTML for the interface, but interactive things like React can be used.

14) From all five required points, I have implemented all at its full potential.
15) Get for '/' is used to render the login page
16) Post for '/' is used to match login details and redirect to broadcast page (general welcome page) if details match the input field and remain on the same page and clear data of fields if details do not match.
17) Get for '/grselect' is used to render broadcast page along with user id of who logged in and requested that page.
18) Get for '/:grps/:grpnames' is used to get the id of the user who requested that page in the form of grpnames, and grps is the id of the actual requested page. It redirects to "/:grps" along with user id.
19) Note we can use jquery instead of this Url cascading method.
20) We have to use the synchronous function to maintain their integrity.
21) Get for '/:grps' is used to render group pages or individual chat pages
22) Note individual user page is of form 0_x_y, here x<y and represent two user id of which chat begins
23) Note group individual page is of form 1_x, here x is group id
24) Note we can create a new room on the broadcast page only.
25) Initial1 socket event is used to join a room
26) Initial2 is used for sending record data
27) Initial 3 and 4 are used to send the group and list of people data.
28) toserver is used to get chat from client
29) toclient is used to send data to the client
30) togrpr is used to add new group data in client and server
31) fromimgsend is used to send media data to the server
32) toimgsend is used to send media data to clients immediately
33) my programme is able to send chats immediately as well as store them locally and globally with database and chat with individuals as well as in group and send media
34) It may not work correctly in case of heavy traffic due to a lack of synchronization and security breaches.

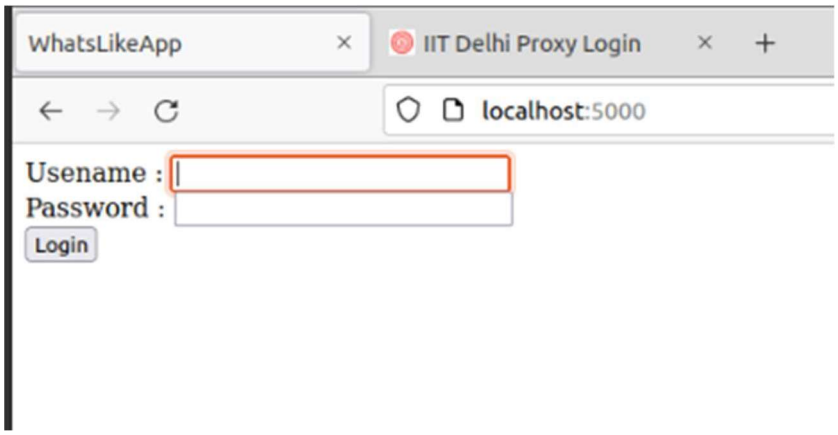


Fig1. Login Page

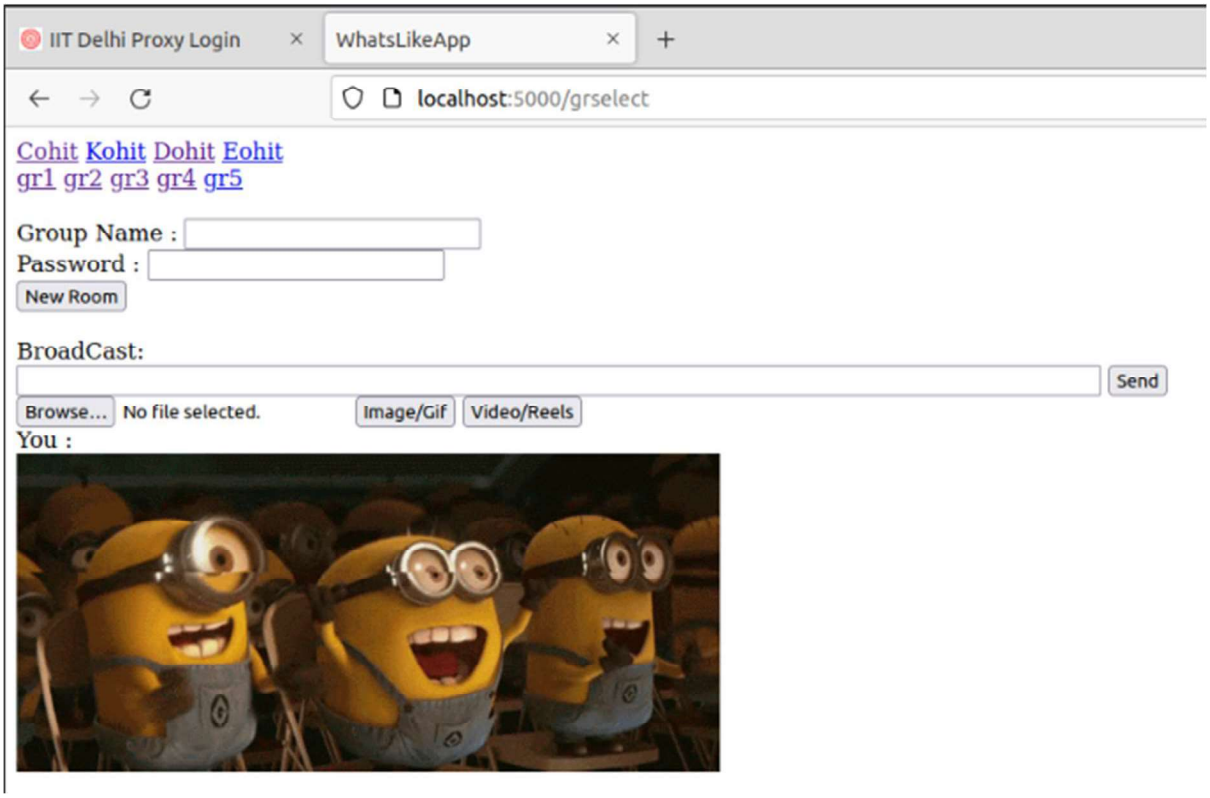


Fig2. Broadcast Page

# Future Scope

1) Better interface
2) Added functionality of back button logout create a new user and other links
3) Better security and load managing by creating variables specific to the instance
4) Integrate more managble database