

Dharmsinh Desai University, Nadiad
Faculty of Technology
Department of Computer Engineering

B. Tech. CE Semester – V
Subject: (CE – 520) Advanced Technologies

Project Title: Chat Application

Group Members: -

Index	Name	Roll No	Permanent ID
1	Umraniya Pritesh Amrutbhai	CE144	21CEUBG070
2	Vataliya Yash Rajeshkumar	CE148	21CEUBG140

Guided By :- Prof. Jigneshkumar Shah

Contents :-

Abstract :-

- Chat application is a feature or a program on the Internet to communicate directly among Internet users who are online or who were equally using the internet. Chat applications allow users to communicate even though from a great distance. Therefore, this chat application must be real-time and multi platform to be used by many users. The development of information and communication technologies are rapidly making one of the reasons for Indonesia, especially Bandung to develop this chat application. That's because Indonesia does not always rely on outsiders. It is important for Indonesia to develop this chat application for themselves. This chat application in the manufacture begins with the collection of relevant data that will be displayed in the web and mobile versions. The programming language used to build server is Node.js with express framework and MongoDB database.

Introduction :-

- The purpose the developing the Chat Application that enables people to communicate with each other in real-time through text messaging. People can communicate anywhere in the world by sending and receiving the messages using the web or mobile app.

SRS document of Chat Application

1. Introduction

1.1 Purpose

1.2 Intended Audience and Reading Suggestions

1.3 Product Scope

1.4 References

2. Overall Description

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

3. External Interface Requirements

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

3.4 Communications Interfaces

4. System Features

5. Other Nonfunctional Requirements

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Responsiveness

5.5 Database

1. Introduction: -

1.1. Purpose: -

The purpose the developing the Chat Application that enables people to communicate with each other in real-time through text messaging. People can communicate anywhere in the world by sending and receiving the messages using the web or mobile app.

1.2. Intended Audience and Reading Suggestions

:-

a) Administrator: -

- Assign users and control access by creating organizational units and groups.
- Managing the data of both admins and users.

b) Users: -

- Created own group or can able to use one to one communication with friends, family.
- Chat application can be use to create group which is helpful for student and professor/teacher to communicate with multiple students.
- Business professionals can also use this to communicate with their clients and customers.

c) Educational institutions : -

- Teachers ad students may use whatsapp to communicate about class schedules, assignments and other education matters.

1.3. Product Scope: -

Chat Application is to offer real-time text based communication application that supports the sending and receiving the messages to people in a faster way. The application also supports the creation of group in which multiple people can communicate with each other.

1.4. References: -

- 1) IEEE - *IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications*, Computer Society, 1998.
- 2) <https://medium.com/@gil.palikaras/build-a-chat-web-app-with-mern-stack-and-socket-io-8f33d151c62c>
- 3) <https://reactjsexample.com/chat-application-built-with-mern-stack/>

2. Overall Description: -

2.1. Product Perspective: -

It will make easier for teachers to carry out learning activities, the intended learning is not only in class but also outside the classroom because student can learn whenever and wherever by accessing classroom online.

The main purpose of the study to investigate the role of classroom in English language teaching.

Sharing material and answering of student doubts become easy and time saving.

2.2. Product Functions: -

- The purpose of this document is to make system more practical, teacher can provide study material, providing assignments, take quiz etc.
- Student should also submit the assignments.
- Teachers can share the notices and schedules.
- Student can ask their doubt in doubt room of respective classroom.
- Student and teachers can create the classroom or join the classroom.
- Teachers can add students in the classroom.

2.3. User classes and Characteristics: -

1) Administrator: -

- Sign up for google classroom for education to enable classroom.
- Assign users and control access by creating organizational units and groups.
- Verify teachers for added functionalities.
- Change user roles to ensure users are identified correctly.
- Set up permissions for your domain to determine who can create class.
- Enable google for enhanced collaboration.

2) Teachers: -

- Created own classroom providing related details.
- They can add student or teachers to the classroom.
- They used to create and manage assignments give marks to students according to their work.
- They can solve students doubts related to the study.
- Post announcements on the class stream.
- Set up and join a meet video call.
- Schedule posts across multiple classes at once.

3) Students: -

- They can join the classroom by class code of respective classroom.
- Students can access assignments, read it but they don't allow to modify the content.
- Submit option is available and also the deadline of work.
- They can get all the study materials.
- Give the opinion which are asked in it.
- They can ask doubts and get reply from other students or teacher.
- They can use doubt session room to ask the doubt to teachers and students.

2.4. Operating Environment: -

Our system can run on different browsers such as Chrome, Mozilla Firefox, Microsoft Edge, Opera.

System can run on different platforms.

2.5. Design and Implementation Constraints: -

This system uses different types of programming language and language framework such as HTML, CSS, JavaScript, Node.js, React.js, Express.js, Angular.js and MongoDB for database and Socket.io for bidirectional communication between clients and servers.

2.6. User Documentation: -

This system will provide online help to solve any difficulty which user face while using application and we try to solve user query.

3. External Interfaces Requirements: -

3.1. User Interfaces: -

It allows teachers and students to access and interact with the various features and tools of the system. These can include a dashboard for managing classes and assignments, tools for communicating with students and other teachers and resources for delivering and tracking student progress.

3.2. Hardware Interfaces: -

It includes physical equipment the physical device and equipment that are used to support teaching and learning activities. These includes laptops, or tablets that are used by teachers and students to access online resources and complete assignments. It also includes Projectors, Interactive white boards and audio equipment.

3.3. Software Interfaces: -

It includes following properties :-

Dashboard: a centralized view that allows teachers to manage classes, assignments and student progress.

Calender: A view of upcoming classes and assignments that allows teachers and students to plan and organize their work.

Class management: tools for creating and managing classes and adding or removing students and set up assignment and quizzes.

Communication: a messaging system that allows teachers and students to communicate with one other either in real time or asynchronously.

3.4. Communication Interfaces: -

A messaging system that allows teachers and students to communicate with one another, either in real-time or asynchronously. This can be done through text, audio or video messages and may include features such as read receipts and message threads.

An email system that allows teachers and students to send and receive messages and attachments to each other.

Chat rooms: a tool for creating and moderating real time chat rooms where students can communicate with one another and with the teacher.

4. System Features: -

Functional Requirements: -

1. User Authentication: -

R1: User should be able to create account on the system and his information should be secure within this system and he should be able to login or logout from the system through this account.

R1.1 display login/sign-in option

Input: choose login for already created account users and sign in for new users.

Output: User enter necessary details in this form.

R1.2 submit details to server

Input: choose submit button

Output: all details of the user submitted to the server. Now user

will be redirect to the home page of the system.

2. User Profile: -

R2: User should be able to create and manage their profiles and he should have ability to customize their profile settings and privacy preferences.

R2.1: User should view or edit the profile.

Input: choose profile option

Output: Profile page is displayed

R2.2 edit profile option is displayed

Input: choose edit profile option

Output: new box is displayed and it will ask for new information to be changed with old one.

R2.3 enter information

Input: enter modified field values and choose save option

Output: now all information in the database will be modified with this information.

3. Contact Management: -

User should be able to search for and add other users as contacts and it should also provide options to manage contacts, such as grouping or categorizing them and should also display option for block the user or delete.

R3.1 : add user for chat, delete user

Input: choose add option.

Output: display to enter mobile number.

R3.2

Input: enter the mobile number.

Output: now that person chat is saved in system and now user can easily chat with that person

4. Real time messaging:

User should be able to send and receive text messages in real time. And it should support both one-on-one and group conversation. And user can check other user online or offline status.

R4: User send messages to his friend and his friend can see that message and give reply to that message or react to that message

Input:

Display box to enter the message.

Output: now it will be sent to that person , notification will be displayed on that person system ,it can be seen from his system when he reads then blue tick is shown

R4.2 Now user can react to that message with other message or emoji

Input : display box for reply to that message or react to that message with emoji.

Output: now this reply is shown in that person system.

5. Message Notification: -

→The application should provide real time notification for incoming messages.

→User should have option to enable or disable notification for specific chats or contacts.

→Notifications can be in the form of sound alerts, pop up messages or push notifications on mobile devices.

6. Sending files

→User should be able to send documents in any format like docs, pdf, excel or any other type.

R6.1 : System display input box to upload file.

Input: Enter multiple files into the system.

Output: now all files are uploaded to the server and it will be sent to that person.

R6.2 : all files are displayed in other user system he can download that file.

Input: display box to react that message with emoji or message.

Output: that reply will be shown to that first user.

7. Message search and filtering: -

User should be able to search for specific messages within a chat or across multiple chats.

The system should provide filtering options to sort and display the messages based on various criteria.

R7: display box to search or filter messages.

Input: enter message for search or filter message based on various criteria.

Output: based on that criteria message will be categorized.

8. Security and privacy: -

→ User should have control over privacy setting including the ability to block or report abusive users.

→ The system should implement mechanisms to prevent unauthorized access to user accounts and conversations.

5. Other Non-functional Requirements :-

5.1. Performance Requirements: -

This system will work on different platforms and web browsers without slowing down the speed and it gives the faster way to use the Chat Application.

5.2. Security Requirements: -

Other person cannot modify details of the other users. No one can change the password of users. information of one chats will not change by other person not are in that group. no one can modify the details which is related with system.

5.3. Safety Requirements: -

This system should follow the rules an regulation of government. Details of each users which are in the Chat application is secure.

5.4. Responsiveness: -

This application show the perfect and layout of the chat application in different devices such as mobile, tablets etc. All the contains clearly visible to the users.

5.5. Database: -

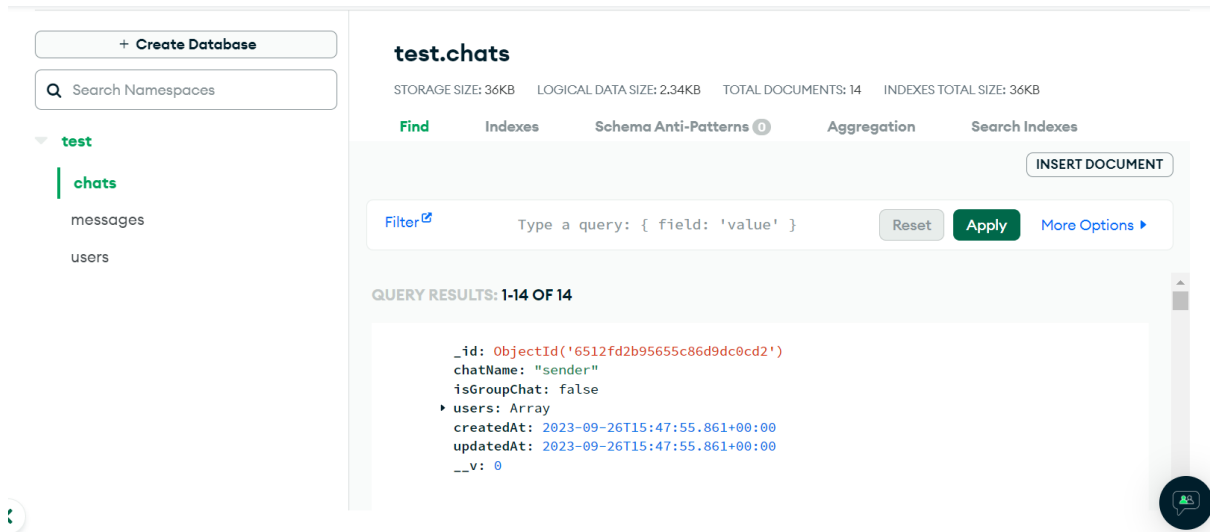
The database is maintained in such way so that the faster managing of the data. And this system database is normalized to remove inconsistency and ambiguity.

6. Goals of Implementation: -

- Provide the notifications of the chat to the user and the update and new features of chat application.
- Ask users to give feedback for this system.
- Enhancing Users and business professionals engagement and participation to use the facility of chat application.

Database Design :-

- Chats collections contains `_id`, `chatName`, `isGroupChat`, `users`, `createdAt`, `updatedAt` fields.



test.chats
STORAGE SIZE: 36KB LOGICAL DATA SIZE: 2.34KB TOTAL DOCUMENTS: 14 INDEXES TOTAL SIZE: 36KB

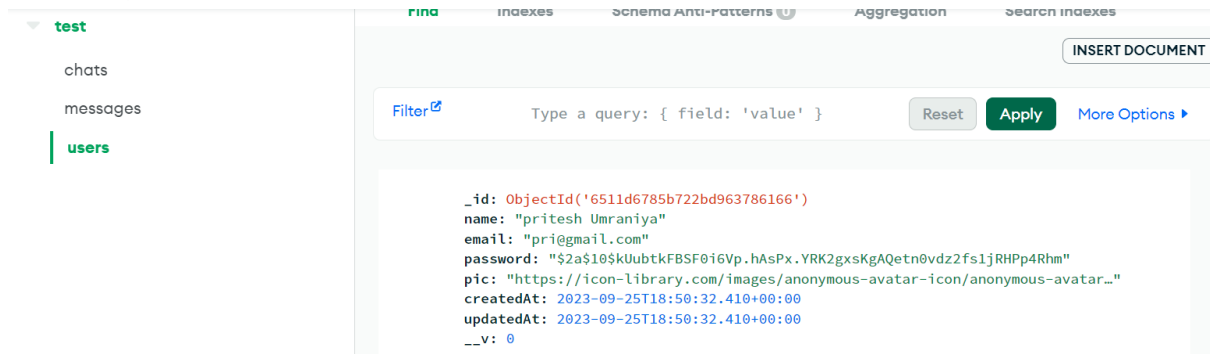
Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Filter Type a query: { field: 'value' } Reset Apply More Options

QUERY RESULTS: 1-14 OF 14

```
{
  "_id": ObjectId('6512fd2b95655c86d9dc0cd2'),
  "chatName": "sender",
  "isGroupChat": false,
  "users": Array,
  "createdAt": 2023-09-26T15:47:55.861+00:00,
  "updatedAt": 2023-09-26T15:47:55.861+00:00,
  "__v": 0
}
```

- User collections has fields `_id`, `name`, `email`, `password`, `createdAt`, `updatedAt`.



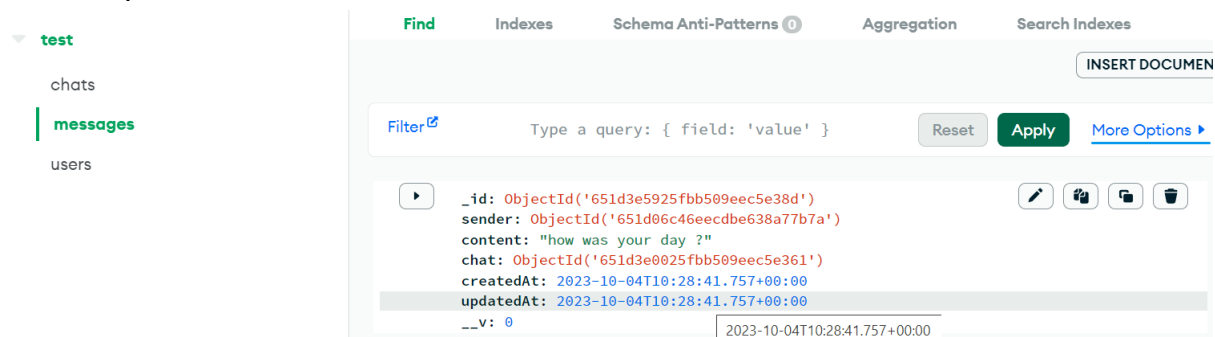
test.users
STORAGE SIZE: 36KB LOGICAL DATA SIZE: 2.34KB TOTAL DOCUMENTS: 14 INDEXES TOTAL SIZE: 36KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Filter Type a query: { field: 'value' } Reset Apply More Options

```
{
  "_id": ObjectId('6511d6785b722bd963786166'),
  "name": "pritesht Umranika",
  "email": "pri@gmail.com",
  "password": "$2a$10$KubtkFBSF0i6Vp.hAsPx.YRK2gxskGAQetn0vdz2fs1jRHPp4Rhm",
  "pic": "https://icon-library.com/images/anonymous-avatar-icon/anonymous-avatar...",
  "createdAt": 2023-09-25T18:50:32.410+00:00,
  "updatedAt": 2023-09-25T18:50:32.410+00:00,
  "__v": 0
}
```

- Messages collection has fields `_id`, `sender`, `content`, `chat`, `createdAt`, `updatedAt`.



test.messages
STORAGE SIZE: 36KB LOGICAL DATA SIZE: 2.34KB TOTAL DOCUMENTS: 14 INDEXES TOTAL SIZE: 36KB

Find Indexes Schema Anti-Patterns Aggregation Search Indexes

Filter Type a query: { field: 'value' } Reset Apply More Options

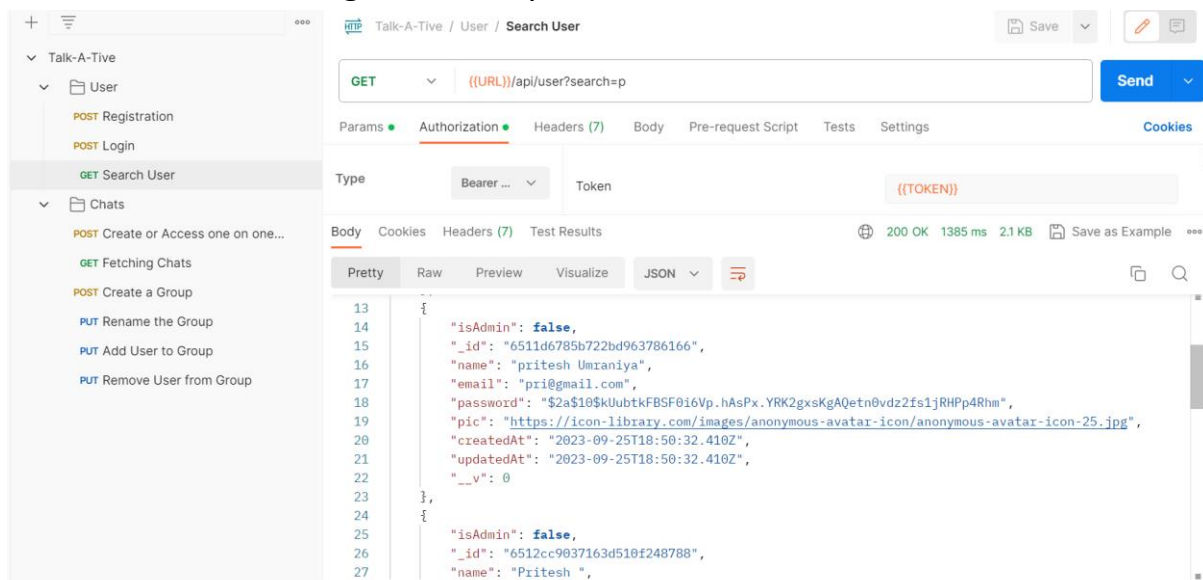
```
{
  "_id": ObjectId('651d3e5925fbb509eec5e38d'),
  "sender": ObjectId('651d06c46eecdbe638a77b7a'),
  "content": "how was your day ?",
  "chat": ObjectId('651d3e0025fbb509eec5e361'),
  "createdAt": 2023-10-04T10:28:41.757+00:00,
  "updatedAt": 2023-10-04T10:28:41.757+00:00,
  "__v": 0
}
```

Implementation Detail :-

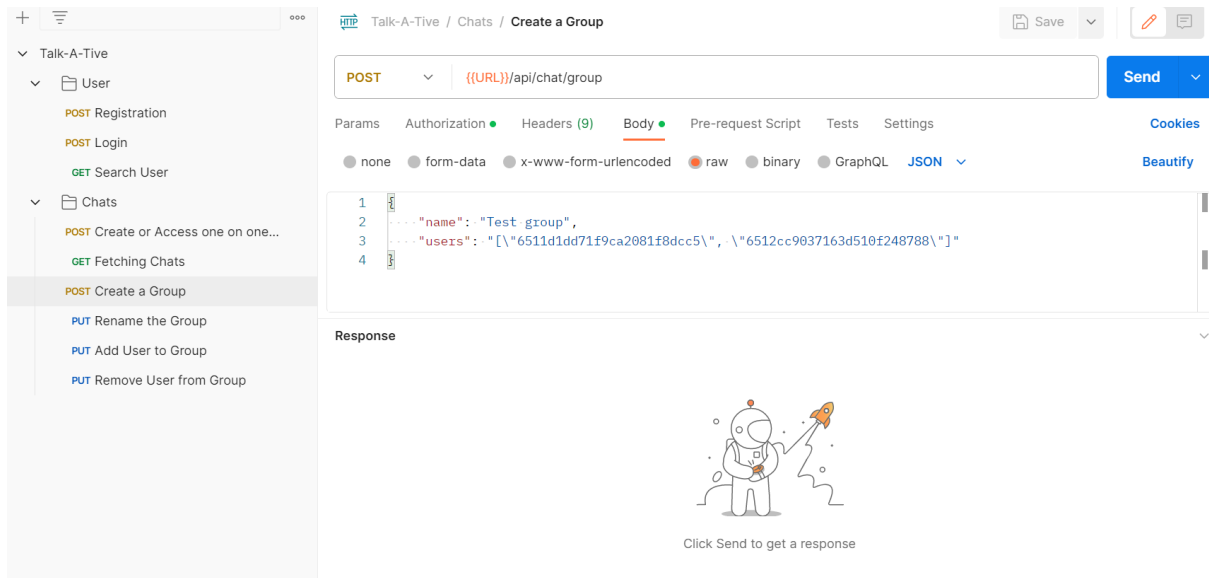
- There are total three collections for chat application system – Users, chats, messages
- To manage user details and there chat details for each collections there is router.js file and controller.js file and model.js like – for chat – chatModel.js, chatControllers.js, chatRoutes.js .
- For User we have created three APIs one is for register and login and searching the user.
- For Chats and Messages there are total 6 APIs created to handle creation of single to single chat, group chat, add person to the group, remove the person from the group, delete the group.

Testing :-

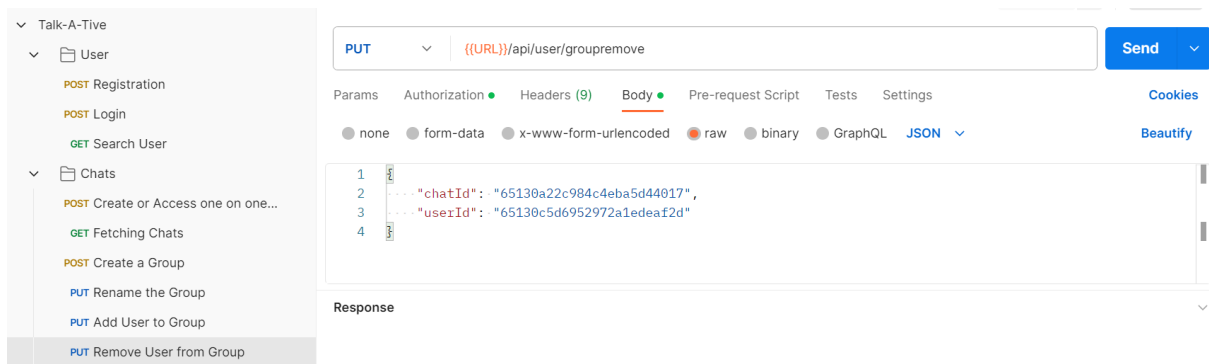
- API for searching the User by name.



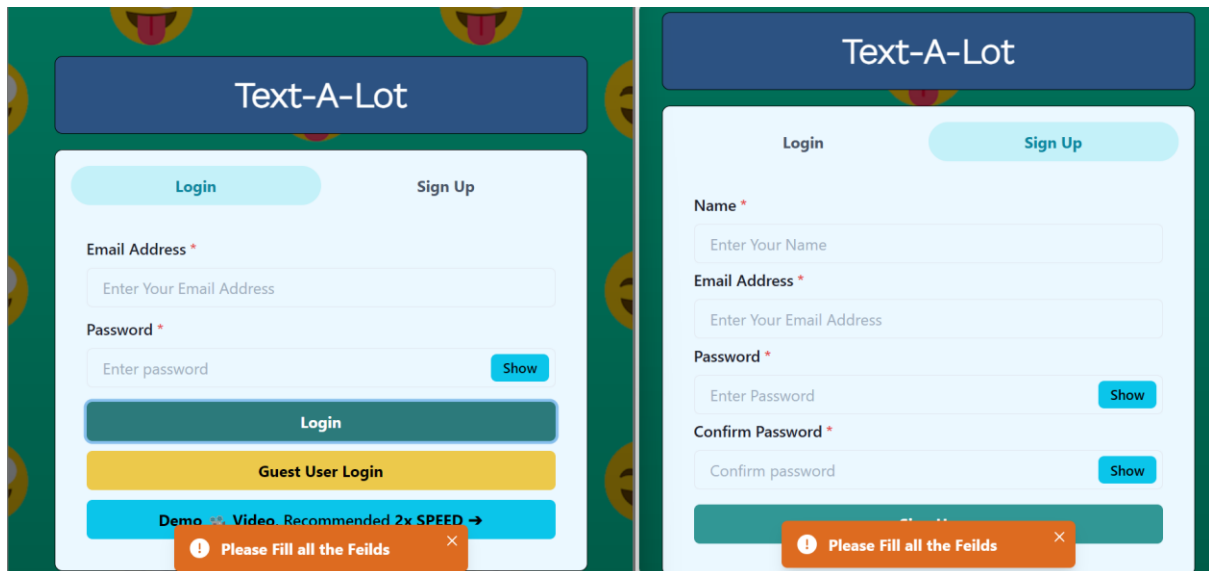
- API for creating the group .



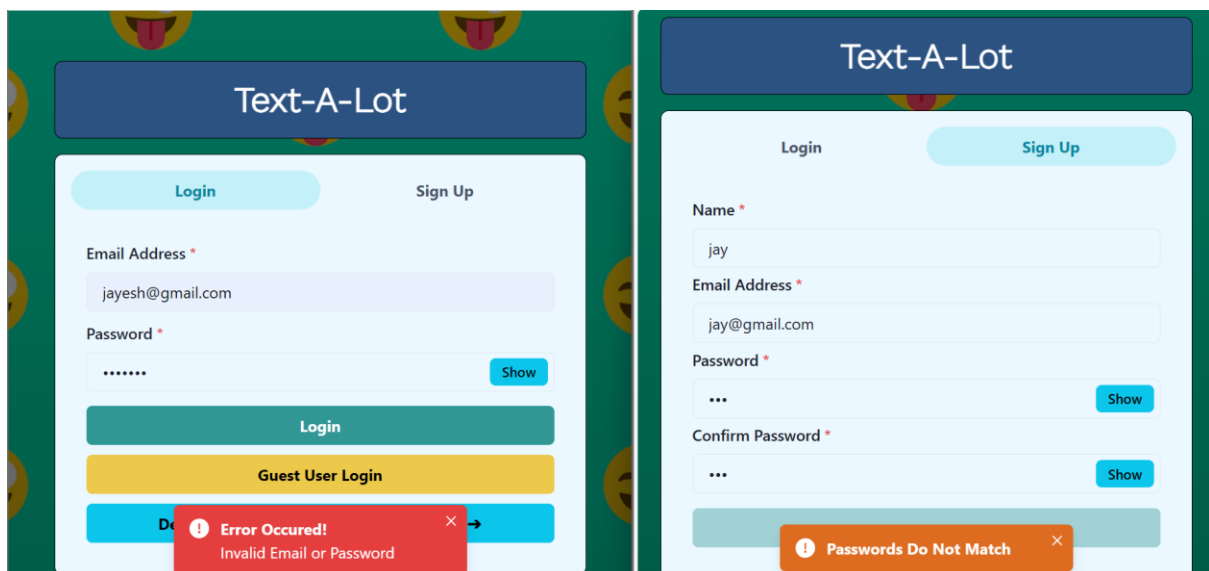
➤ API for removing the person from the group.



➤ We have created APIs in the Postman to test the system and also there are some other testing shown below while user working with the system.

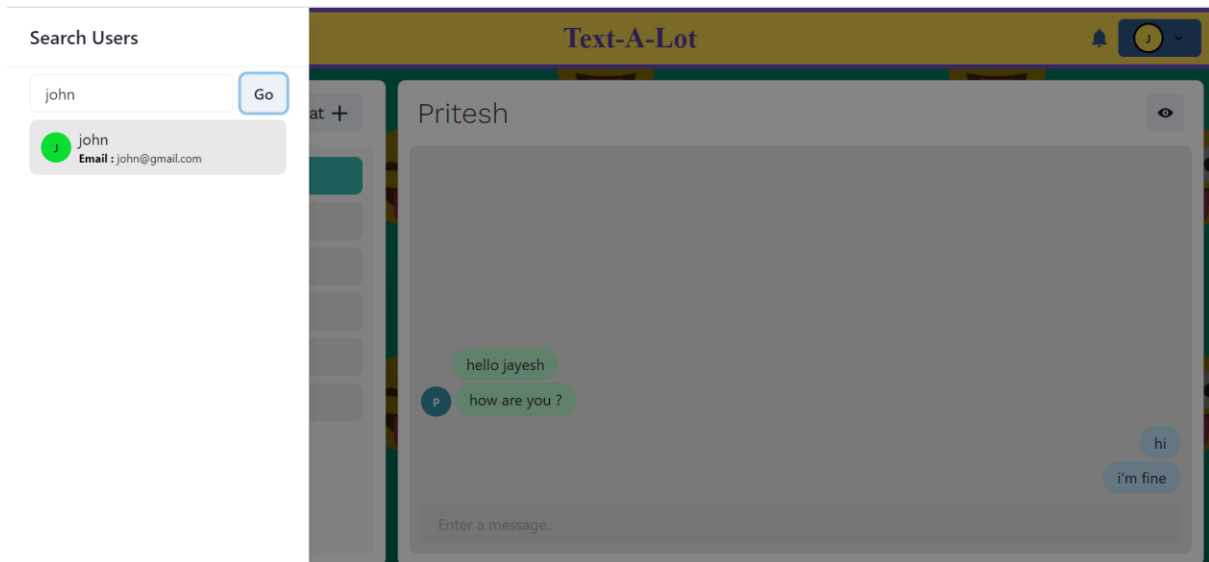


➤ Password validation.

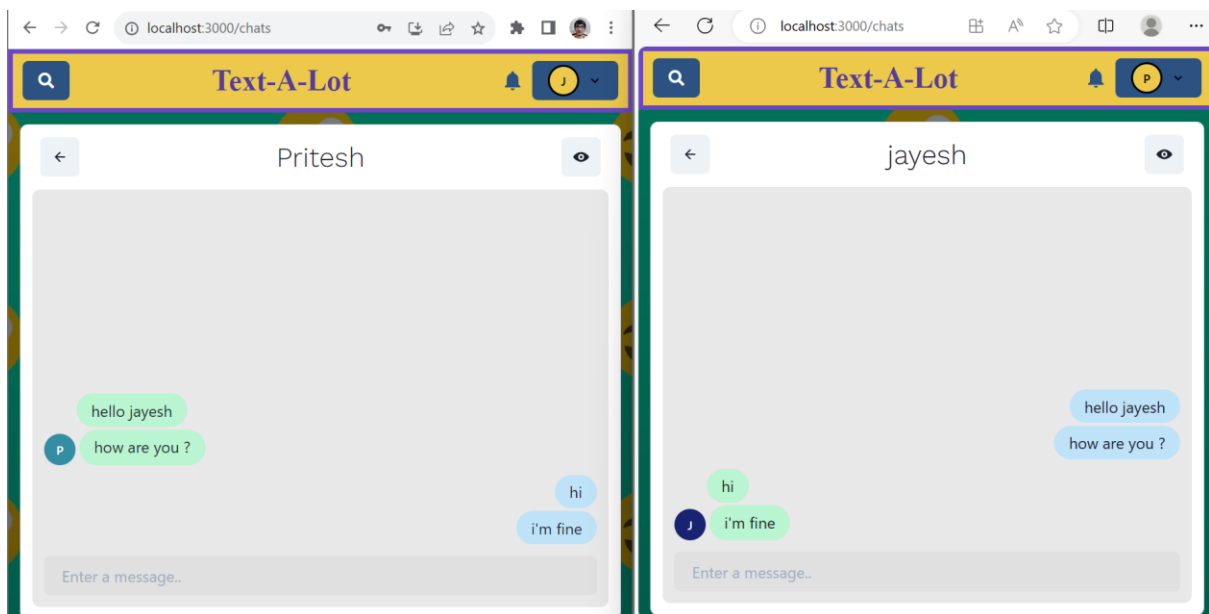


Screenshots of important features :-

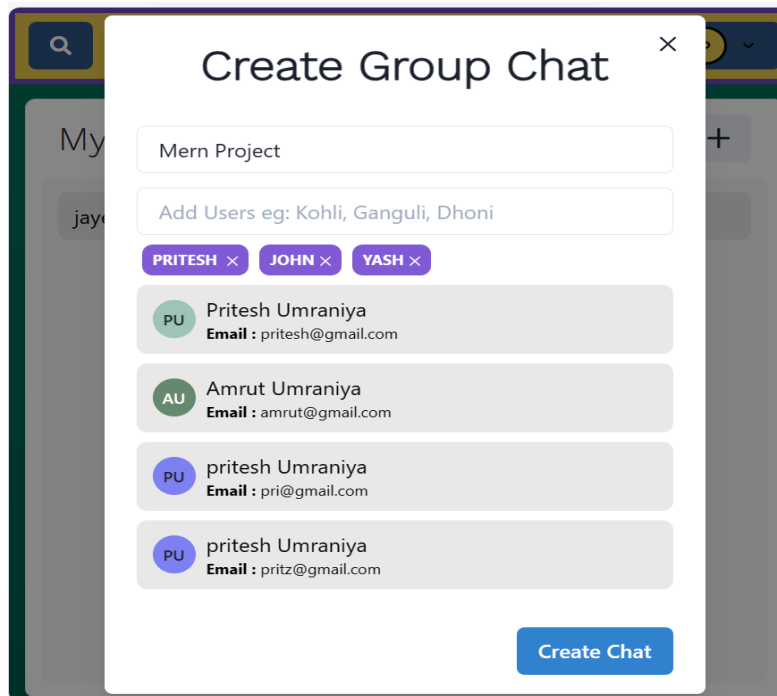
➤ Search s user by name.



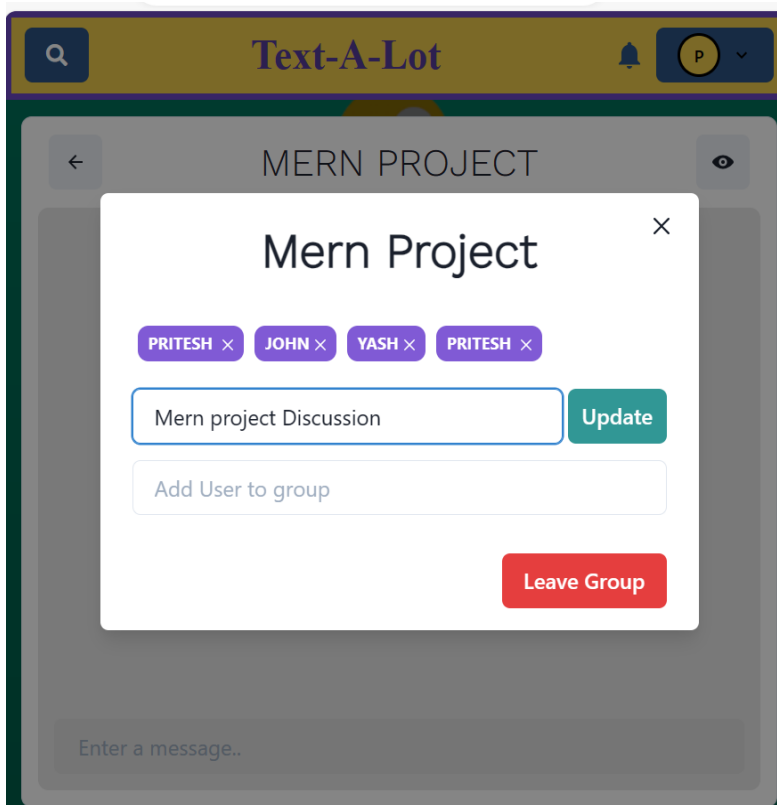
- User can create a single chat and as well as Group chat (The person who create group chat is the Considered as Admin). Send messages and emojis.



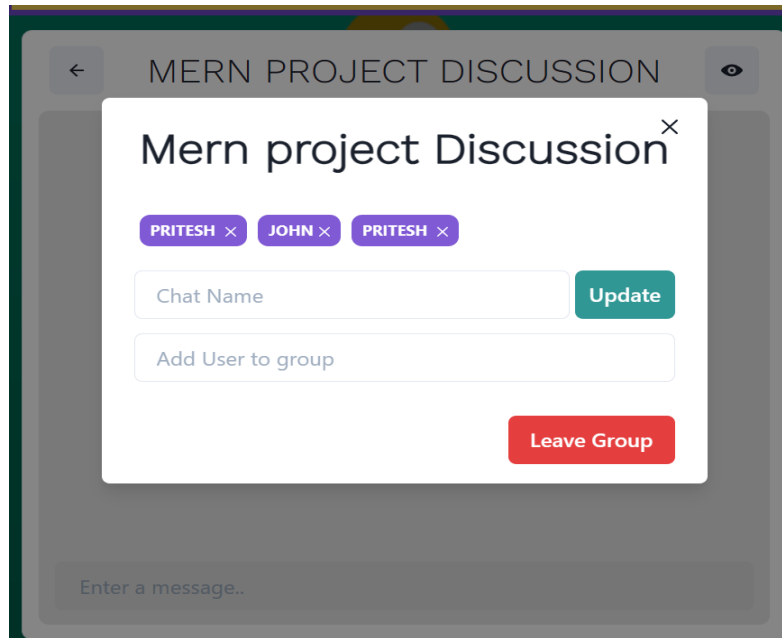
- Create the group chat and add people into them.



- Update the group (changing group name 'Mern project' to 'Mern project disucssion' (only admin can do this)).



- Delete the person from the group (only admin can do this). And leave the group feature.



Conclusion :-

- In this application, User can send message, emojis to other users so there is bidirectional communication made by using socket.io.
- User can also create group chat between two people and can add, update or delete multiple users from the group chat but only have access for that functionality not any user from the group chat.

Limitation and Future Extension :-

- limitation of this project when many users do chat with different users on the system then all should connect to same database so there is chance such that server might be crash in such situation due to handling many requests at the same time.
- In this application users can send messages and also create group chat but not able to send documents or send status which will disappear after 24 hours.
- Our future plan is to create this type of functionality and also create quiz, poll in this system.

Bibliography :-

- <https://www.mongodb.com/atlas/database>
- <https://chakra-ui.com/>
- <https://fontawesome.com/icons>
- <https://socket.io/docs/v4/tutorial/introduction>
- <https://react.dev/learn>
- <https://nodejs.org/en>
- <https://expressjs.com/>
- <https://jwt.io/>
- <https://www.mongodb.com/products/tools/compass>
- <https://www.npmjs.com/package/nodemon>
- <https://hackernoon.com/a-guide-on-writing-tests-in-full-stack-mern-web-application>