MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Project Report

on

CHAT APPLICATION

A project report submitted in partial fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Mranalini Joshi

0901CS191060

Pallavi

0901CS191074

Faculty Mentor:

Ms. Khushboo Agrawal

Assistant Professor, Computer Science and Technology

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE

GWALIOR - 474005 (MP) est. 1957

MAY-JUNE 2022

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

Project Report

on

CHAT APPLICATION

A project report submitted in partial fulfillment of the requirement for the degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

Submitted by:

Mranalini Joshi

0901CS191060

Pallavi

0901CS191074

Faculty Mentor:

Mrs. Khushboo Agrawal

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE

GWALIOR - 474005 (MP) est. 1957

MAY-JUNE 2022

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR
(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

CERTIFICATE

This is certified that **Mranalini Joshi** (0901CS191060) has submitted the project report titled Chat Application under the mentorship of **Ms. Khushboo Agrawal**, in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering from Madhav Institute of Technology and Science, Gwalior.

[illegible]

Ms. Khushboo Agrawal
Faculty Mentor
Assistant Professor
Computer Science and Engineering

[illegible] 09/05/20[illegible]

Dr. Manish Dixit,
Professor and Head,
Computer Science and Engineering

Dr. Manish Dixit
Professor & HOD
Department of CSE
M.I.T.S. Gwalior

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

CERTIFICATE

This is certified that **Pallavi** (0901CS191074) has submitted the project report titled Chat Application under the mentorship of **Ms. Khushboo Agrawal**, in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in Computer Science and Engineering from Madhav Institute of Technology and Science, Gwalior.

[illegible]

Ms Khushboo Agrawal
Faculty Mentor
Assistant Professor
Computer Science and Engineering

[illegible] 09/05/2022

Dr. Manish Dixit,
Professor and Head,
Computer Science and Engineering

Dr. Manish Dixit
Professor & HOD
Department of CSE
M.I.T.S. Gwalior

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

DECLARATION

I hereby declare that the work being presented in this project report, for the partial fulfilment of requirement for the award of the degree of Bachelor of Technology in Computer Science and Engineering at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of Ms Khushboo Agrawal, Professor, Computer Science and Technology.

I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

[illegible]

Mranalini Joshi
0901CS191060
III Year,
Computer Science and Engineering

Pallavi

Pallavi
0901CS191074
III Year,
Computer Science and Engineering

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE, GWALIOR

(A Govt. Aided UGC Autonomous & NAAC Accredited Institute Affiliated to RGPV, Bhopal)

ACKNOWLEDGEMENT

The full semester project has proved to be pivotal to my career. I am thankful to my institute, **Madhav Institute of Technology and Science** to allow me to continue my disciplinary/interdisciplinary project as a curriculum requirement, under the provisions of the Flexible Curriculum Scheme (based on the AICTE Model Curriculum 2018), approved by the Academic Council of the institute. I extend my gratitude to the Director of the institute, **Dr. R. K. Pandit** and Dean Academics, **Dr. Manjaree Pandit** for this.

I would sincerely like to thank my department, **Department of Computer Science and Engineering, for allowing** me to explore this project. I humbly thank **Dr. Manish Dixit**, Professor and Head, Department of Computer Science and Engineering, for his continued support during the course of this engagement, which eased the process and formalities involved.

I am sincerely thankful to my faculty mentors. I am grateful to the guidance of **Ms Khushboo Agrawal**, Professor, Computer Science and Technology, for ~~his~~ her continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

[illegible]

Mranalini Joshi
0901CS191060
III Year,
Computer Science and Engineering

Pallavi

Pallavi
0901CS191074
III Year,
Computer Science and Engineering

TABLE OF CONTENTS

ABSTRACT

Clients Chatting is a method of using technology to bring people and ideas together despite geographical barriers. The technology has been available for years but the acceptance was quite recent. Our project is an example of a chat server. It is made up of two applications - the client application, which runs on the user's web browser, and the server application runs on any hosting servers on the network. To start chatting clients should get connected to a server where they can do private and group chats. Security measures were taken during the last one.

Keyword: Server, Client, Broadcasting, Communication, Application, UI Component

सार :

क्लाइंट चैटिंग भौगोलिक बाधाओं के बावजूद लोगों और विचारों को एक साथ लाने के लिए प्रौद्योगिकी का उपयोग करने का एक तरीका है। तकनीक वर्षों से उपलब्ध है लेकिन स्वीकृति काफी हाल ही में हुई थी। हमारा प्रोजेक्ट चैट सर्वर का एक उदाहरण है। यह दो अनुप्रयोगों से बना है - क्लाइंट एप्लिकेशन, जो उपयोगकर्ता के वेब ब्राउज़र पर चलता है, और सर्वर एप्लिकेशन नेटवर्क पर किसी भी होस्टिंग सर्वर पर चलता है। चैटिंग शुरू करने के लिए क्लाइंट्स को एक सर्वर से कनेक्ट होना चाहिए जहां वे प्राइवेट और ग्रुप चैट कर सकें। पिछले एक के दौरान सुरक्षा उपाय किए गए थे।

LIST OF TABLES

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTON

Chat applications play an important role in how the world interacts today, due to their immediacy and vast capabilities. A chat application makes it easy to communicate with people anywhere in the world by sending and receiving messages in real-time. With a chat app, users are able to receive the same engaging and lively interactions through custom messaging features, just as they would in person. This also keeps users conversing on your platform instead of looking elsewhere for a messaging solution. Whether it's a private chat, group chat, or large scale chat, adding personalized chat features to your app can help ensure that your users have a memorable experience.

1.2 PROBLEM STATEMENT

- This project is to create a chat application with a server and user to enable the user to chat with each other's.
- To develop an instant messaging solution to enable users to seamlessly communicate with each other.
- The project should be very easy to use enabling even a novice person to use it.
- This project can play an important role in an organizational field where employees can connect through LAN.
- The main purpose of this project is to provide multi-chatting functionality through a network.

1.3 Innovative Ideas of Project

• **GUI**: Easy to use GUI (Graphical User Interface), hence any user with minimal knowledge of operating a system can use the software.

•**Platform independence**: The messenger operates on any system irrelevant of the underlying operating system.

•**Unlimited clients**: "N" number of users can be connected without any performance degradation of the server.

1.4 Project Objective

•Communication: To develop an instant messaging solution to enable users to seamlessly communicate with each other.

•User friendliness: The project should be very easy to use enabling even a novice person to use it.

1.5 Scope of The Project

•Broadcasting Chat Server Application is going to be a text communication software, it will be able to communicate between two computers using point-to-point communication.

•The limitation of Live Chat is it does not support audio conversations. To overcome this limitation, we are concurrently working on developing better technologies.

•Companies would like to have communication software wherein they can communicate instantly within their organization.

•The fact that the software uses an internal network set up within the organization makes it very secure from outside attacks.

1.6 What is React?

- React.js is an open-source JavaScript library that is used for building user interfaces specifically for single-page applications.
- It's used for handling the view layer for web and mobile apps.
- React also allows us to create reusable UI components.
- React allows developers to create large web applications that can change data, without reloading the page.
- The main purpose of React is to be fast, scalable, and simple. It works only on user interfaces in the application. This corresponds to the view in the MVC template.

1.6.1 Why use React?

Data Binding:

React uses one-way data binding and an application architecture called Flux controls the flow of data to components through one control point – the dispatcher. It's easier to debug self-contained components of large ReactJS apps.

1. *Performance*:

React does not offer any concept of a built-in container for dependency. You can use Browserify, Require JS, EcmaScript 6 modules which we can use via Babel, ReactJS-di to inject dependencies automatically.

2. *Testability*:

ReactJS applications are super easy *1. Simplicity*:

ReactJS is just simpler to grasp right away. The component-based approach, well-defined lifecycle, and use of just plain JavaScript make React very simple to learn, build a professional web (and mobile applications), and support it. React uses a special syntax called JSX which allows you to mix HTML with JavaScript. This is not a requirement; the Developer can still write in plain JavaScript but JSX is much easier to use.

3. *Easy to learn*:

Anyone with a basic previous knowledge of programming can easily understand React while Angular and Ember are referred to as 'Domain-specific Language', implying that it is difficult to learn them. To react, you just need basic knowledge of CSS and HTML.

4. *Native Approach*:

React can be used to create mobile applications (React Native). And React is a diehard fan of reusability, meaning extensive code reusability is supported. So at the same time, we can make IOS, Android, and Web applications.

to test. React views can be treated as functions of the state, so we can manipulate with the state we pass to the ReactJS view and take a look at the output and triggered actions, events, functions, etc.

1.7 What is socket.io?

- Socket.IO is a library that enables **low-latency**, **bidirectional** and **event-based** communication between a client and a server.
- It is built on top of the WebSocket protocol and provides additional guarantees like fallback to HTTP long-polling or automatic reconnection.
- It works on every platform, browser or device, focusing equally on reliability and speed. Socket.IO is built on top of the WebSockets API (Client side) and Node.js. It is one of the most depended upon library on **npm** (Node Package Manager).

1.7.1 Features of socket.io

Here are the features provided by Socket.IO over plain WebSockets:

- reliability (fallback to HTTP long-polling in case the WebSocket connection cannot be established)
- automatic reconnection
- packet buffering
- acknowledgments
- broadcasting to all clients or to a subset of clients (what we call "Room")
- multiplexing (what we call "Namespace")

CHAPTER 2: PRODUCT OVERVIEW

The functionality of the chat application is to give the ability to chat with whoever is online on the application. The users and stakeholders will be a small group, for now, the use cases will be what is available to the user, and the functional/non-functional requirements will be covered, as well as the milestones of the chat application.

2.1 Users and Stakeholders

This section will deal with the users and stakeholders. The users will be using the chat application and the stakeholders will develop, maintain, and test the chat application.

Team and I	We will be developing, maintaining, and testing the chat application through its phases of development.
Users	The users will be anyone who has the chat application and registers for it.

Table 2.1:Users and Stakeholder

2.2 Project Perspective:

•The system to be developed here is a Chat facility. It is a centralized system. It is Client-Serversystem with centralized database server. All local clients are connected to the centralized servervia LAN.

•There is a two-way communication between different clients and server. This chat applicationcan be used for group discussion. It allows users to find other logged in users.

2.3 Interface:

•This application interacts with the user through G.U.I. The interface is simple, easy to handleand self-explanatory.

•Once opened, user will easily come into the flow with the application and easily uses allinterfaces properly. However, the basic interface available in our application(fig.2) is: -

•Title panel

•Message panel

•Contact list panel

•Status panel

2.4 Functional and Non-Functional Requirements: -

2 .4.1 Functional Requirements

2.User Registration: User must be able to register for the application through an Email, Username and Password. On Opening the application, user must be able to register themselves or they can directly login if there have an account already. If user skips this step, user should able to chat. The user's email will be the unique identifier of his/her account on Chat Application.

3.Adding New Contacts: The application should detect all contacts from the server database. If any of the contacts have user entered with Chat Application, those contacts must automatically be added to the users contact list on Chat Application.

4.Send Message: User should be able to send instant message to any contact on his/her Chat Application contact list. User should be notified when message is successfully delivered to the recipient by coloring message.

5.Broadcast Message: User should be able to create groups of contacts. User should be able to broadcast messages to these groups.

2.4.2 Non-Functional Requirements

1.**Privacy:** Messages shared between users should be encrypted to maintain privacy.

2.**Robustness**: In case user's system crashes, a backup of their chat history must be stored on remote database servers to enable recoverability.

3.**Performance**: Application must be lightweigh and must send messages instantly.

2.5 Use Case Table

Level 0	Level 1	Level 2	Actor
Chat Application	Authentication System	Registrar, Login, Logout	User And Admin
	Contact Form	Search/Find User, Adding User	User
	Chat Form	Send Message	User
	Monitor	Chat History	User and Admin

Table 2.2: Use Case

2.5.1 Use case Diagram

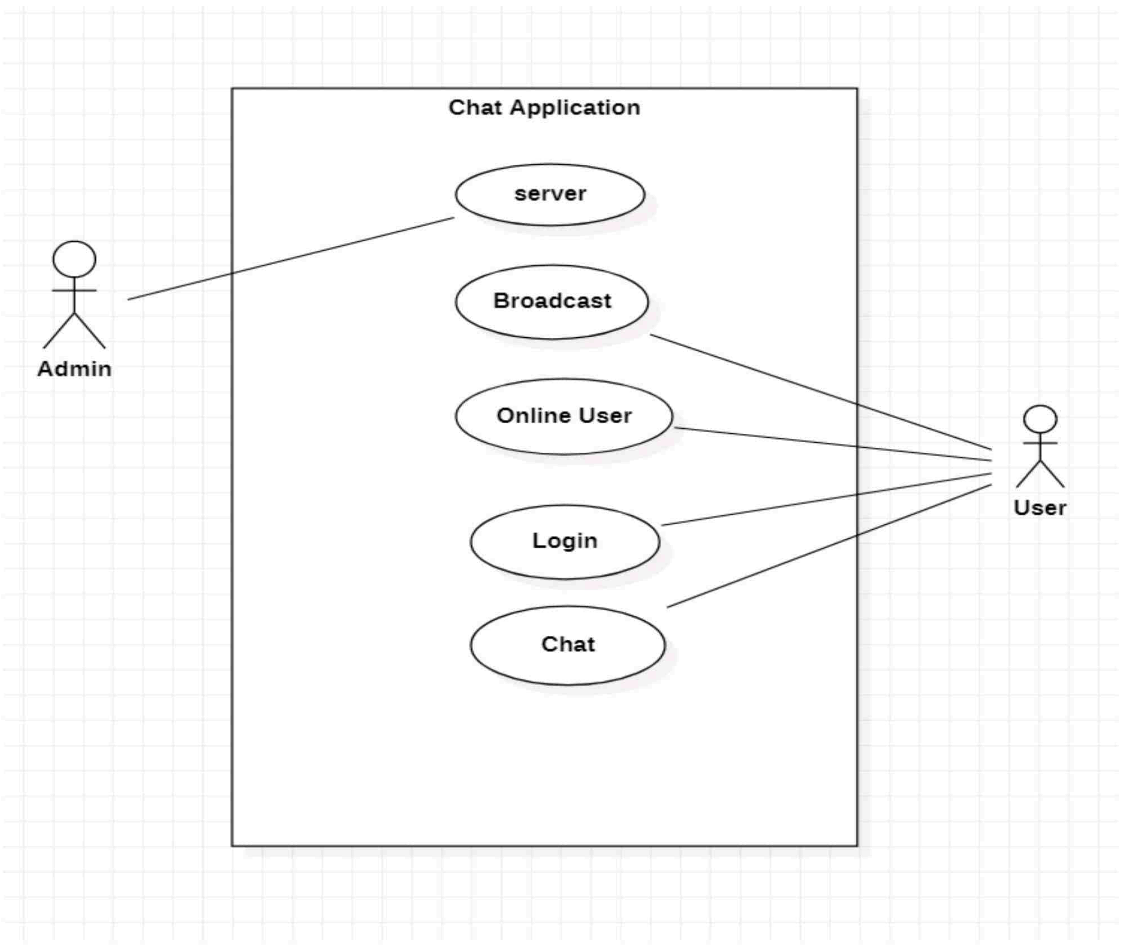
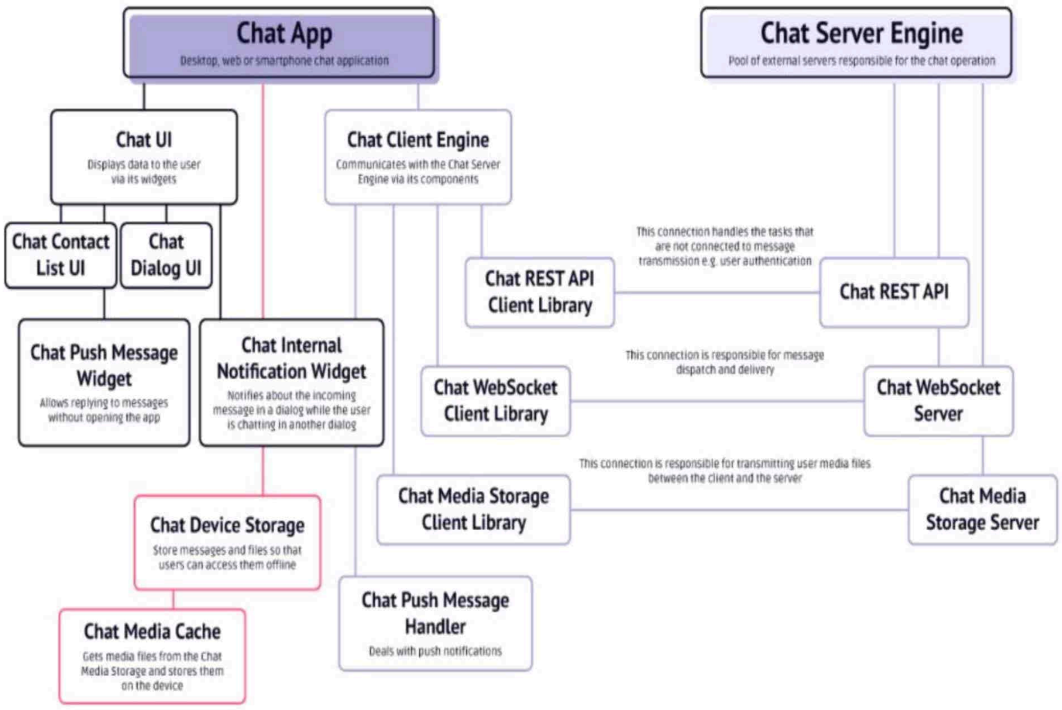


Fig 2.1: Use Case Diagram

CHAPTER 3: ARCHITECTURE

3.1 Chat Architecture: How We Approach It



Chat App
Desktop, web or smartphone chat application
Chat Server Engine
Pool of external servers responsible for the chat operation
Chat UI
Displays data to the user via its widgets
Chat Client Engine
Communicates with the Chat Server Engine via its components
Chat Contact List UI
Chat Dialog UI
This connection handles the tasks that are not connected to message transmission e.g. user authentication
Chat REST API Client Library
Chat REST API
Chat Push Message Widget
Allows replying to messages without opening the app
Chat Internal Notification Widget
Notifies about the incoming message in a dialog while the user is chatting in another dialog
This connection is responsible for message dispatch and delivery
Chat WebSocket Client Library
Chat WebSocket Server
This connection is responsible for transmitting user media files between the client and the server
Chat Device Storage
Store messages and files so that users can access them offline
Chat Media Storage Client Library
Chat Media Storage Server
Chat Media Cache
Gets media files from the Chat Media Storage and stores them on the device
Chat Push Message Handler
Deals with push notifications

Figure 3.1 Architecture

A chat consists of two major parts:

•Chat App or client part, which is a Web chat application. Build on React

•Chat Server Engine or server part, which is a pool of external servers responsible for the chatoperation. This is the place where all the chat magic happens.

Both parts contain various components that communicate to each other and bring the chat into action.

3.2 Chat App or Client Side

Chat App is the other major part of the chat architecture, the one that users directly interact with. It's split into two separate root components:

• Chat Client Engine handles all the communication with the Chat Server Engine via its internal components: Chat REST API Client Library and Chat WebSocket Client Library.

•Chat UI displays data to users: Chat Contact List UI Chat Dialog UIComponent:

Components are the building blocks of any React app and a typical React app will have many of these. Simply put, a component is a JavaScript class or function that optionally accepts inputs i.e. properties(props) and returns a React element that describes how a section of the UI (User Interface) should appear.App.js is the starting point of our React app.

•lists the packages your project depends on

•specifies versions of a package that your project can use.

•makes your build reproducible, and therefore easier to share with other developers

3.3 Chat Server Engine

This is a core of the chat architecture that handles message delivery and dispatch. In our version of chat architecture, it includes the following components:

•**Chat REST API** handles the tasks that are not connected directly to message dispatch and delivery, such as user authentication, changing of user settings, friends invitation, downloading sticker packs, etc. The Chat App (the chat client part) communicates with the Chat REST API via the Chat REST API Client Library

•**Chat WebSocket Server** is responsible for transmitting messages between users. The Chat App communicates with the Chat WebSocket Server via the Chat WebSocket Client Library. This connection is open two ways; that means users don’t have to make requests to the server if there are any messages for them, they just get them right away.

The third-party package or modules installed using npm are specified in this segment. The package.json file is the heart of the Node.js system. It is the manifest file of any Node.js project and contains the metadata of the project. The package.json file is the essential part to understand, learn and work with theNode.js. It is the first step to learn about development inNode.js.

Chapter 4: Final Analysis and Design

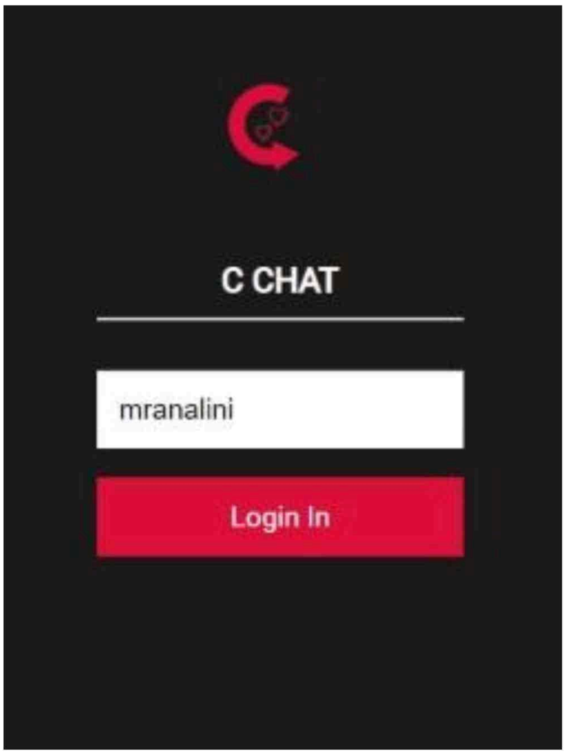
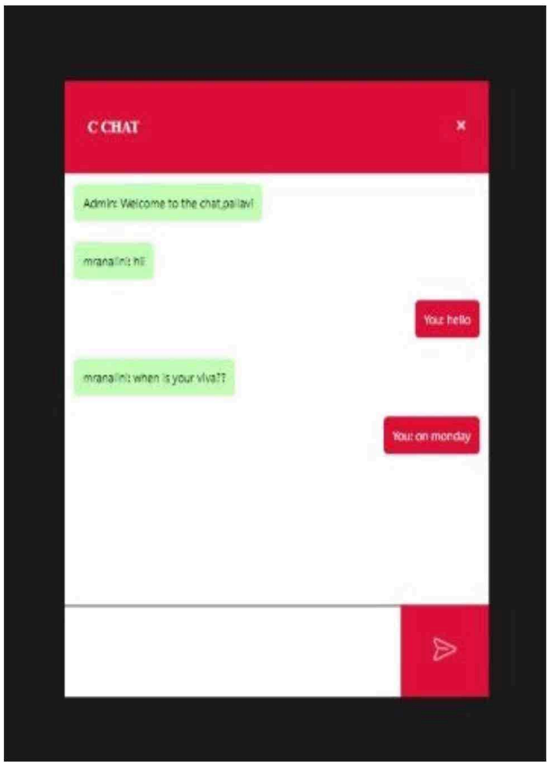


Fig.4.1 Screenshot



Admin: Welcome to the chat,pallavi

mranalini: hi!

You: hello

mranalini: when is your viva??

You: on monday

Fig.4.2 Screenshot

4.1 Problem Faced

- The server was not be reachable
- There was be an issue with the SSL certificate of the server

4.2 Limitations

- If server is not in running condition then you can not start the chatting.
- This application is used for sending and receiving the messages but the message is shown to every client. Means this application is unable to provide the private chat facility it is like a multi party conference.

Conclusion

There is always a room for improvements in any app. Right now, we are just dealing with text communication. There are several chat apps that serve similar purpose as this project, but these apps were rather difficult to use and provide confusing interfaces. A positive first impression is essential in the human relationship as well as in human-computer interaction. This project hopes to develop a chat service Web app with high-quality user interface.

Future Work

In future we may be extended to include features such as:

•File Transfer

•Voice Message

•Video Message

•Audio Call

•Video Call

•Group Call

References

1. https://www.digitalocean.com/community/books/how-to-code-in-react-js-ebook
2. https://matfuvit.github.io/UVIT/predavanja/literatura/TutorialsPoint%20node.js.pdf
3. https://socket.io/docs/v4/