MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

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

NAAC Accredited with A++ Grade

Project Report

on

TALKATIVE

Submitted By:

Shishant Katare

0901AI211058

Faculty Mentor:

Prof. Ravi Kumar Jain

CENTRE FOR ARTIFICIAL INTELLIGENCE

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE

GWALIOR - 474005 (MP) est. 1957

JULY-DEC. 2023

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

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

NAAC Accredited with A++ Grade

CERTIFICATE

This is certified that **Shishant Katare (0901AI211058)** has submitted the project report titled TALKATIVE under the mentorship of **Prof. Ravi kumar jain**, in partial fulfilment of the requirement for the award of degree of Bachelor of Technology in **Artificial Intelligence And Robotics** from Madhav Institute of Technology and Science, Gwalior.

[illegible]

Prof. Ravi Kumar Jain
Faculty Mentor

Centre for Artificial Intelligence

[illegible] 23/11/23

Dr. R. R. Singh
Coordinator
Centre for Artificial Intelligence

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

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

NAAC Accredited with A++ Grade

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 **Artificial Intelligence And Robotics** at Madhav Institute of Technology & Science, Gwalior is an authenticated and original record of my work under the mentorship of **Prof. Ravi Kumar Jain**, Centre for Artificial Intelligence. I declare that I have not submitted the matter embodied in this report for the award of any degree or diploma anywhere else.

Shishant

Shishant Katare
0901AI211058
3rd Year,
Centre for Artificial Intelligence

MADHAV INSTITUTE OF TECHNOLOGY & SCIENCE GWALIOR

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

NAAC Accredited with A++ Grade

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, **Centre for Artificial Intelligence,** for allowing me to explore this project. I humbly thank **Dr. R. R. Singh**, Coordinator, Centre for Artificial Intelligence, 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 **Prof. Ravi Kumar Jain** , <Designation>, <Department Name>, for his continued support and guidance throughout the project. I am also very thankful to the faculty and staff of the department.

Shishant

Shishant Katare
0901AI211058
3rd Year,
Centre for Artificial Intelligence

ABSTRACT

With the turn of events and improvement in the web, an ever-increasing number of individuals have been choosing online chat platforms for correspondence. Their applications include being used for correspondence over significant distances. Along these lines, the application must both be continuous and multi-stage, and utilized by numerous clients. The online ongoing talking application need not bother with any extra outsider customer program, and visual correspondence could be set up advantageously. The programming instrument utilized to build this application is React.js, Node.js, which has an express structure and an in-memory data set. The text correspondence is moved to and from servers. Furthermore, the information transmission is achieved through highlight point association between servers. Because of the use of the response structure, a virtual space idea is carried out, which improves the presentation over existing applications.

Keywords: real time; chat; web; message; room; group chat

TABLE OF CONTENTS

1. INTRODUCTION

The Web Chat application is an element or a program on the Internet used to impart information straightforwardly among Internet clients who are on the web or who are similarly utilizing the web. Talk applications permit clients to convey information from a distance. Thus, this talk application should be ongoing and multi-stage in order to be utilized by numerous clients. The improvement of data and correspondence innovations is quickly achieved through in-memory databases. Assembling this application starts with the assortment of significant information that will be shown on the web and versatile adaptations. The programming language used to construct workers is Node.js with express structure . Users can chat in real time in a room by simply signing in and choosing which room they want to join. The client can share messages and files in real time within the room. Internal memory databases will store data that rely primarily on memory for data storage, in contrast with databases that store data on disks or Solid State Device (SSD) In-memory information stores are intended to empower negligible reaction times by eliminating the need for circles. Since all information is put away and overseen solely in the primary memory, in-memory data sets risk losing information upon an interaction or server disappointment . In-memory data sets can maintain information on plates by putting away every activity in a log or by taking pictures.

Features:

- Easy to use: Our platform is easy to use and accessible from any device. Simply create an account, choose a username, and start chatting.
- Public and private chat rooms: You can join public chat rooms or send private messages to individuals.
- Group chats: Chat with multiple people at the same time in our group chat rooms.
- Safe and welcoming environment: We have a strict anti-harassment policy and work hard to ensure that all users feel comfortable and respected.

Benefits:

- Connect with people from all over the world: Talkative allows you to connect with people from all over the world and make new friends.
- Share your thoughts and ideas: Share your thoughts and ideas with others and get their feedback.
- Engage in meaningful discussions: Talkative is a great place to engage in meaningful discussions about a variety of topics.
- Learn about different cultures: Chat with people from different cultures and learn about their way of life.
- Have fun: Talkative is a fun and engaging way to connect with others and spend your time.

TALKATIVE

We hope to see you soon!

Talkative is a great way to connect with people from all over the world and make new friends. It is a fun and engaging way to spend your time and learn about different cultures. With its easy-to-use features and safe and welcoming environment, Talkative is the perfect place for anyone looking to connect with others online

In addition to the benefits listed above, Talkative can also be a valuable tool for businesses. Businesses can use Talkative to connect with their customers, provide customer support, and conduct market research Talkative can also be a great way for businesses to promote their products and services.

If you are looking for a way to connect with people from all over the world, share your thoughts and ideas, and learn about different cultures, then Talkative is the perfect place for you. Sign up today and start chatting!

2. Literature

2.1. Problem Statement

• The aim of this study is to create a talk application with a server and clients to empower clients to call one another

• The study aims to foster a solution to empower clients to flawlessly speak with one another

• The undertaking ought to be exceptionally simple to empower even a novice to utilize it.

• This undertaking can assume a significant role in authoritative fields where representatives can interface through LAN.

• The primary motivation behind this task is to provide multiple useful forms of communication th [illegible] network.

2.2. Project Objective

• Correspondence: to foster a texting platform to empower clients to consistently speak with one another

• Ease of use: The venture ought to be exceptionally simple, allowing even a novice to utilize it

2.3 SCOPE

i. A real-time chat platform that allows users to connect with others from all over the world

ii. A safe and welcoming environment for all users.

iii. Easy to use and accessible from any device.

iv. Offers a variety of features, including public and private chat rooms, group chats.

2.4 Requirements:

i. A back-end server to store user data and handle chat messages.

ii. A front-end web application that allows users to chat with each other.

iii. A mobile app for users to chat on the go.

iv. A moderation team to ensure that the talkative remains a safe and welcoming environment.

2.5 USER REQUIREMENTS

2.5.1 Functional Requirements:

i. **Create and manage user accounts:** Users should be able to create and manage their accounts, including setting up a username, password, and profile information.

ii. **Send and receive messages:** Users should be able to send and receive messages in real-time, both in public chat rooms and private conversations.

iii. **Join and leave chat rooms:** Users should be able to join and leave public chat rooms at will.

iv. **Add and remove contacts**: Users should be able to add and remove contacts from their personal list.

v. **Edit profile information:** Users should be able to edit their profile information, such as their name, bio, and picture.

vi. **Report abuse:** Users should be able to report abusive or inappropriate behavior to moderators.

2.5.2 Non-Functional Requirements:

i. **Performance:** The chat application should be able to handle a large number of concurrent users and messages without experiencing performance issues.

ii. **Security:** The chat application should be secure and protect user data from unauthorized access.

iii. **Usability:** The chat application should be easy to use and navigate, even for those who are not familiar with chat applications.

iv. **Accessibility:** The chat application should be accessible to users with disabilities.

v. **Localization:** The chat application should be available in multiple languages to accommodate a global user base.

These are just a few of the many user requirements for chatting. The specific requirements for a particular chat application will vary depending on the target audience and the specific features of the application

2.6 What Is Express.js?

• Express is an unimportant and adaptable Node.js web application system that gives a hearty arrangement of highlights for web and portable applications. It is an opensource structure created and maintained by the Node.js establishment.

• Express provides the apparatuses that are needed to fabricate our application, be it a single-page, multi-page or crossover web application. It is adaptable, as there are various modules accessible on npm (Node Package Manager), which can be straightforwardly connected to Express.

• Not at all like its rivals, such as Rails and Django, which have an obstinate method of building applications, Express has no "most ideal way" to accomplish something. It is entirely adaptable and pluggable.

• Pug (prior known as Jade) is a succinct language used to compose HTML formats. It produces HTML and upholds dynamic code and code reusability (DRY). It is perhaps the most well-known layout language utilized with Express.

• Express can be considered as a layer based on the highest point of the Node.js that deals with a server and courses. It permits clients to arrange middleware to react to HTTP Requests and characterizes a directing table which is utilized to perform various activities dependent on HTTP strategy and URL.

• Express permits powerful delivery of HTML pages dependent on passing contentions to formats.

• Express is offbeat and single strung and performs I/O tasks rapidly

2.7. What Is React?

• ReactJS is an explanatory, effective, and adaptable JavaScript library for building reusable UI parts. It is an open source, part-based front-end library which is capable just for the view layer of the application. It was at first evolved and maintained by Facebook, and later utilized in applications such as WhatsApp and Instagram.

• A ReactJS application is comprised of different parts, and every part is answerable for yielding a little, reusable piece of HTML code. The parts are the basics of all React applications. These components can be settled with different parts to permit complex applications to be worked on via straightforward structure blocks. ReactJS utilizes a virtual DOM-based system to fill information in HTML DOM. The virtual DOM works quickly as it just changes individual DOM components as opposed to reloading total DOM without fail.

• Rather than utilizing customary JavaScript, React codes are sent in something many refer to as JSX (JavaScript Syntax Extension). JSX is fundamentally a sentence structure augmentation of ordinary JavaScript, and it is utilized to make React components. These components are then delivered to the React DOM. JSX is quicker than typical JavaScript as it performs improvements while using standard JavaScript.

2.8. What Is Node.js?

• Node.js is an exceptionally amazing JavaScript-put together stage that works with respect to Google Chrome's JavaScript V8 Engine. It is utilized to foster I/O escalated web applications such as video web-based locales, single-page applications, and other web applications. Node.js is open source, totally free, and utilized by a large number of engineers all over the planet.

• Node.js is a server-side stage build on Google Chrome's JavaScript Engine (V8 Engine). Node.js was fabricated by Ryan Dahl in 2009

• Node.js applications are written in JavaScript and can be run inside the Node.js runtime on OS X, Microsoft Windows, and Linux.

• Node.js additionally gives a rich library of many JavaScript screens which works on the advancement of web applications utilizing Node.js by and large.

3. IMPLEMENTATION

Creating the full implementation of a CHATTING SYSTEM is a complex task that would require a significant amount of code and multiple files. However, I provided a simplified structure of the implementation using JavaScript with Node.js and Express as a web framework . Socketio as a library and mongodb as a database. Do note that this is a high-level overview, and the actual implementation is quite extensive.

Implementation Steps:

1. Create Directory Structure

i. Organize your project by creating a directory structure to keep files and folders organized.

2. Create an NPM Project and Install Dependencies

i. Set up a Node.js project, and install necessary dependencies and dev dependencies using NPM.

3. Create Express Server

i. Create a server.js file to initialize and configure an Express server.

4. Git Setup and First Commit

i. Initialize a Git repository for version control and make the initial commit.

5. Install Laravel Mix

i. Install Laravel Mix and configure it for managing JavaScript and SCSS.

6. Install React JS

i. Install React JS for building user interfaces

7.Install Chakra UI

i. Install chakra UI for building modern and uses-friendly web application

8. Create Login/Signup page

i. Create Login/Signup page using React and CSS

9. Move Routes to Route Files

i. Organize and move your routes to separate route files for better project structure

10. Create Dedicated Controllers

i. Develop dedicated controller files to manage different parts of your application's logic

11. Add Send and Receive message functionality

12. Create Login Page with Route

i. Develop a login page and set up the associated route.

13. Create Signup page with Route

i. Create Signup page and configure the route.

14. Build Register CRUD

i. Implement registration CRUD operations using Express, incorporating dependencies such as express-flash, express-session, and dotenv

ii. Adhere to status code conventions.

15. Create Table for users

16. Show Profile page

17. create group chat with functionality

18 Show all previous message database

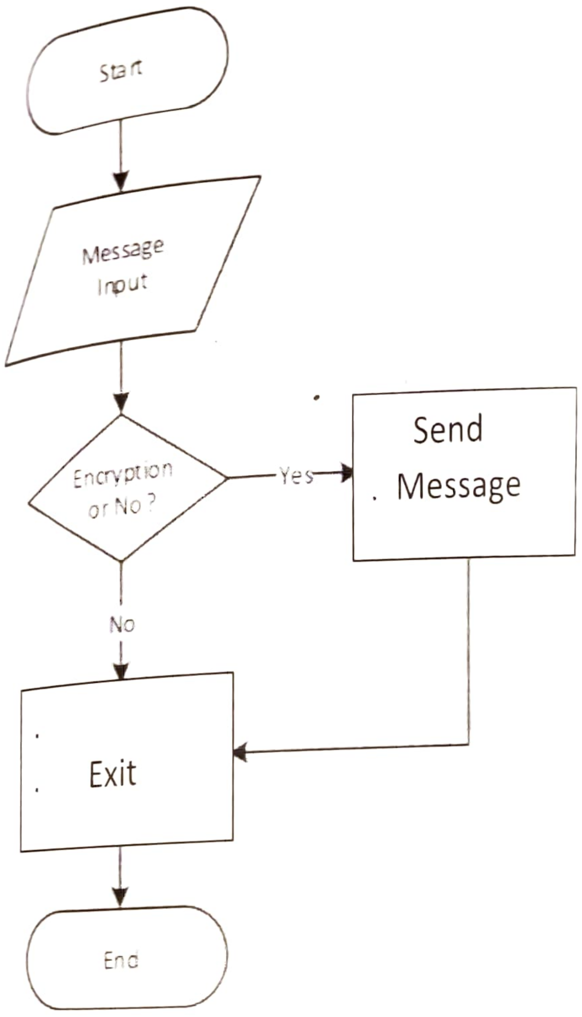
19. Build project for deployment

i. Prepare your project for deployment by optimizing and configuring settings.

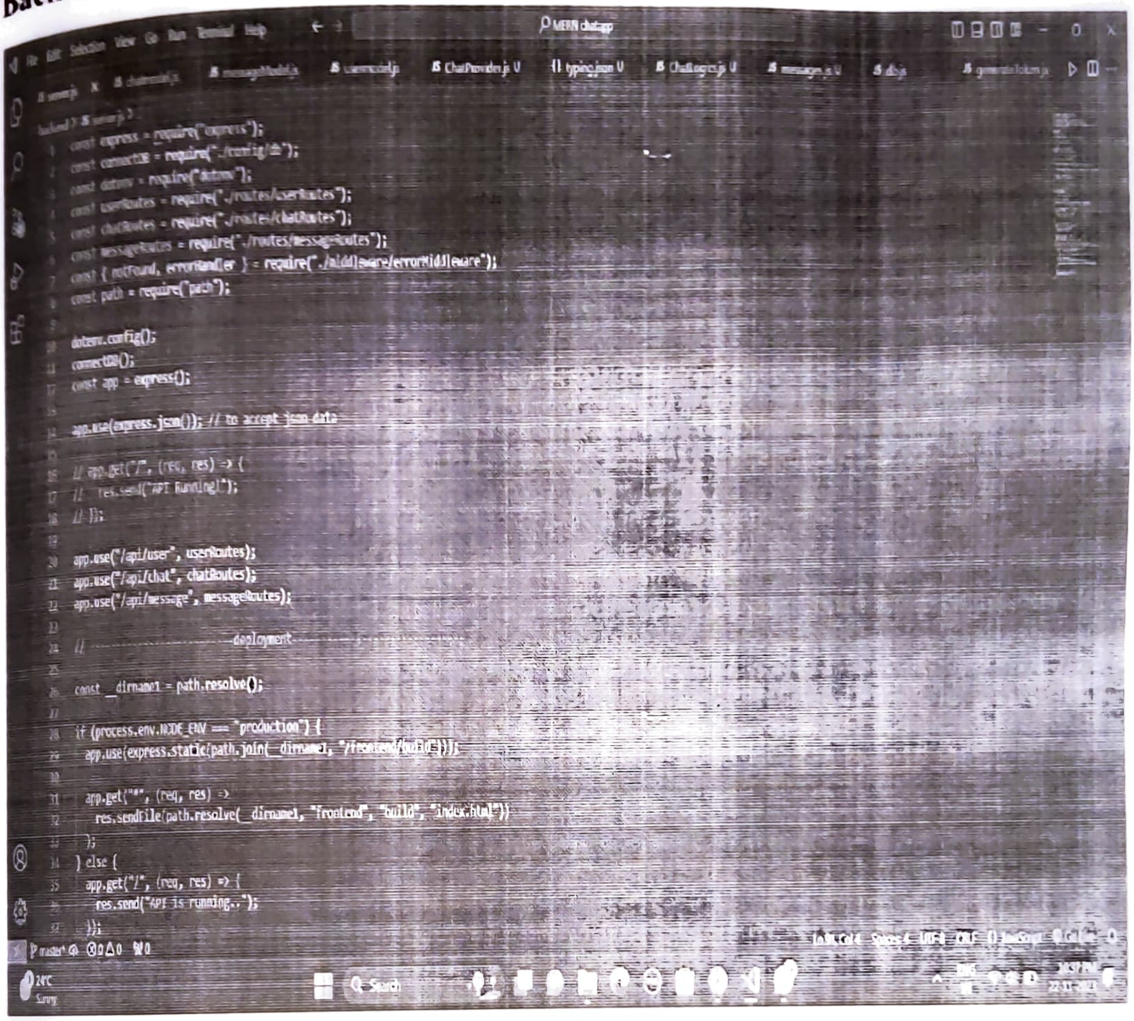
20. Deploy Project on a Live Server

i. Deploy your pizza ordering system on a live server for public access

3.1 FLOWCHART Of Chatting System:

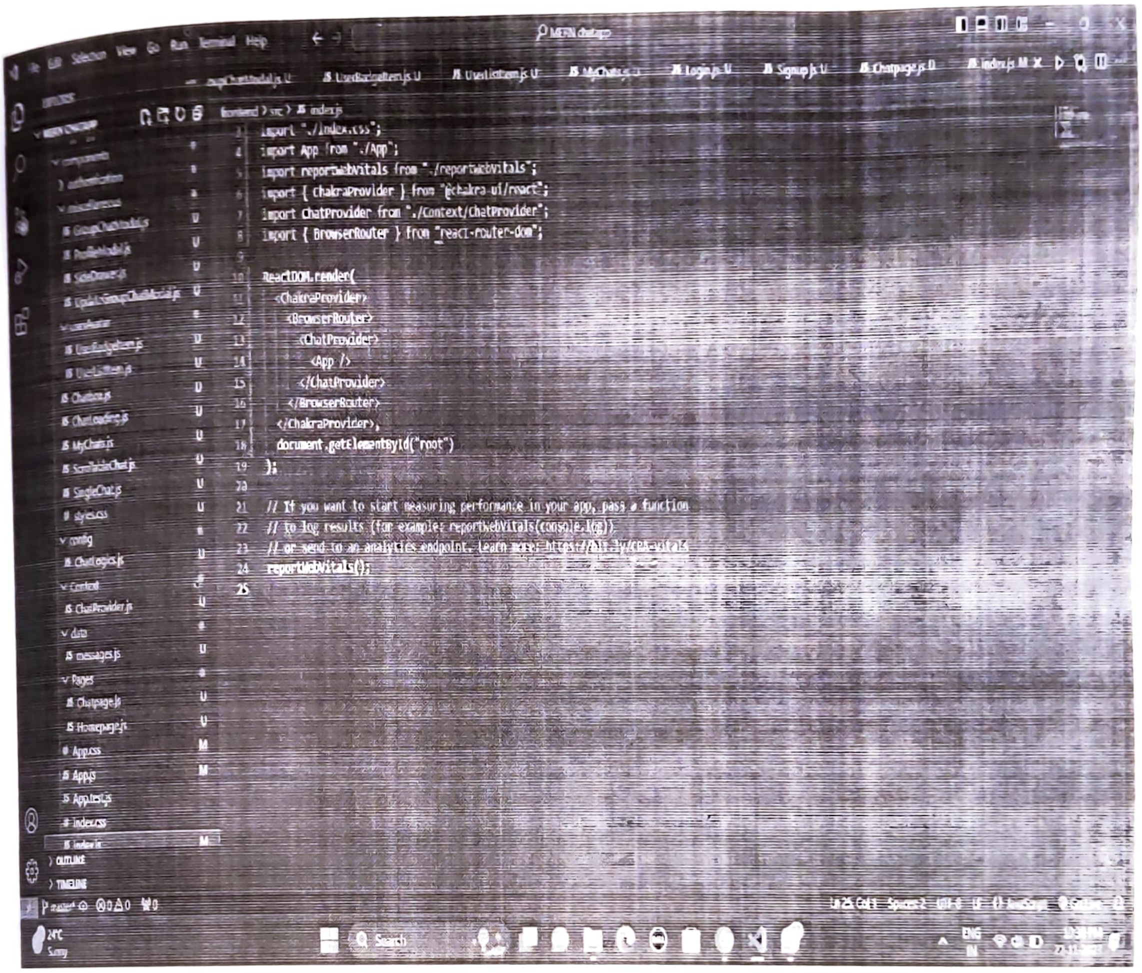


Backend Server.js file



```
const express = require("express");
const connectDB = require("./config/db");
const dotenv = require("dotenv");
const userRoutes = require("./routes/userRoutes");
const chatRoutes = require("./routes/chatRoutes");
const messageRoutes = require("./routes/messageRoutes");
const { notFound, errorHandler } = require("./middleware/errorMiddleware");
const path = require("path");

dotenv.config();
connectDB();
const app = express();

app.use(express.json()); // to accept json data

// app.get("/", (req, res) => {
//   res.send("API Running!");
// });

app.use("/api/user", userRoutes);
app.use("/api/chat", chatRoutes);
app.use("/api/message", messageRoutes);

// --------------------------deployment------------------------------

const __dirname1 = path.resolve();

if (process.env.NODE_ENV === "production") {
  app.use(express.static(path.join(__dirname1, "/frontend/build")));

  app.get("*", (req, res) =>
    res.sendFile(path.resolve(__dirname1, "frontend", "build", "index.html"))
  );
} else {
  app.get("/", (req, res) => {
    res.send("API is running..");
  });
```

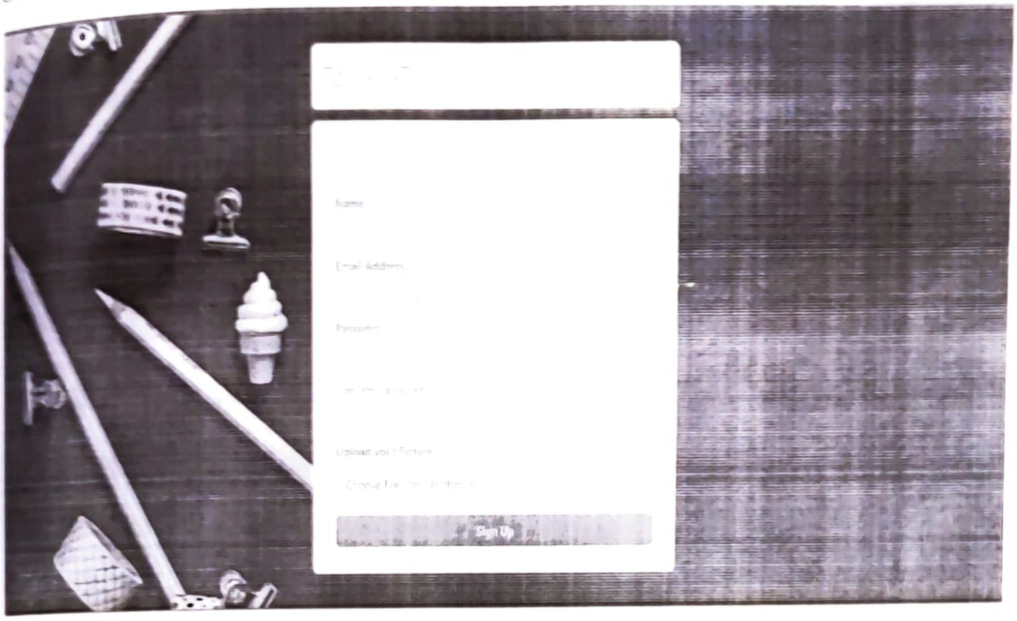
React index.js file



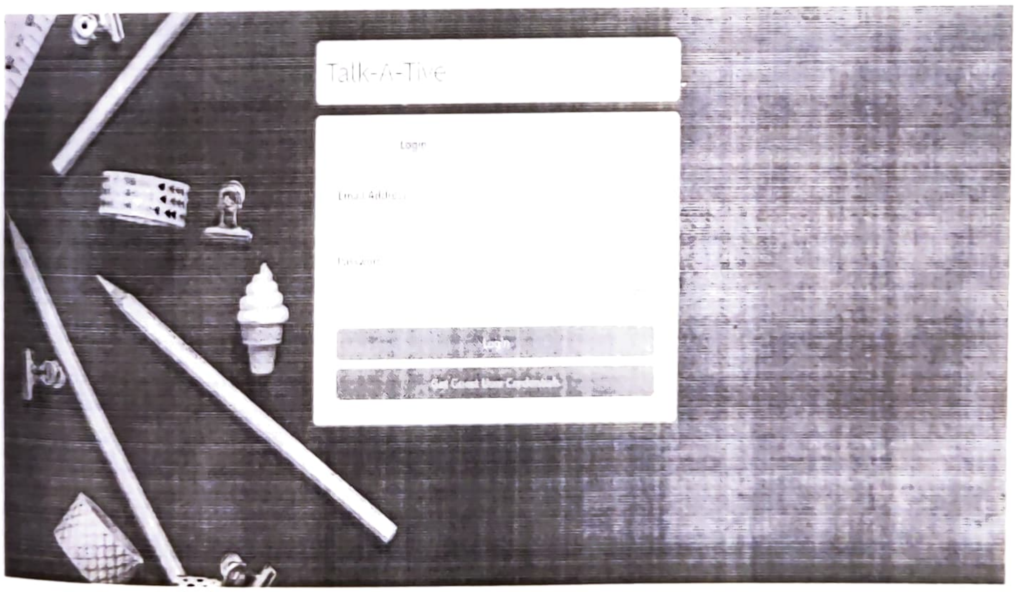
```
import "./index.css";
import App from "./App";
import reportWebVitals from "./reportWebVitals";
import { ChakraProvider } from "@chakra-ui/react";
import ChatProvider from "./Context/ChatProvider";
import { BrowserRouter } from "react-router-dom";

ReactDOM.render(
  <ChakraProvider>
    <BrowserRouter>
      <ChatProvider>
        <App />
      </ChatProvider>
    </BrowserRouter>
  </ChakraProvider>,
  document.getElementById("root")
);

// If you want to start measuring performance in your app, pass a function
// to log results (for example: reportWebVitals(console.log))
// or send to an analytics endpoint. Learn more: https://bit.ly/CRA-vitals
reportWebVitals();
```

4 RESULT

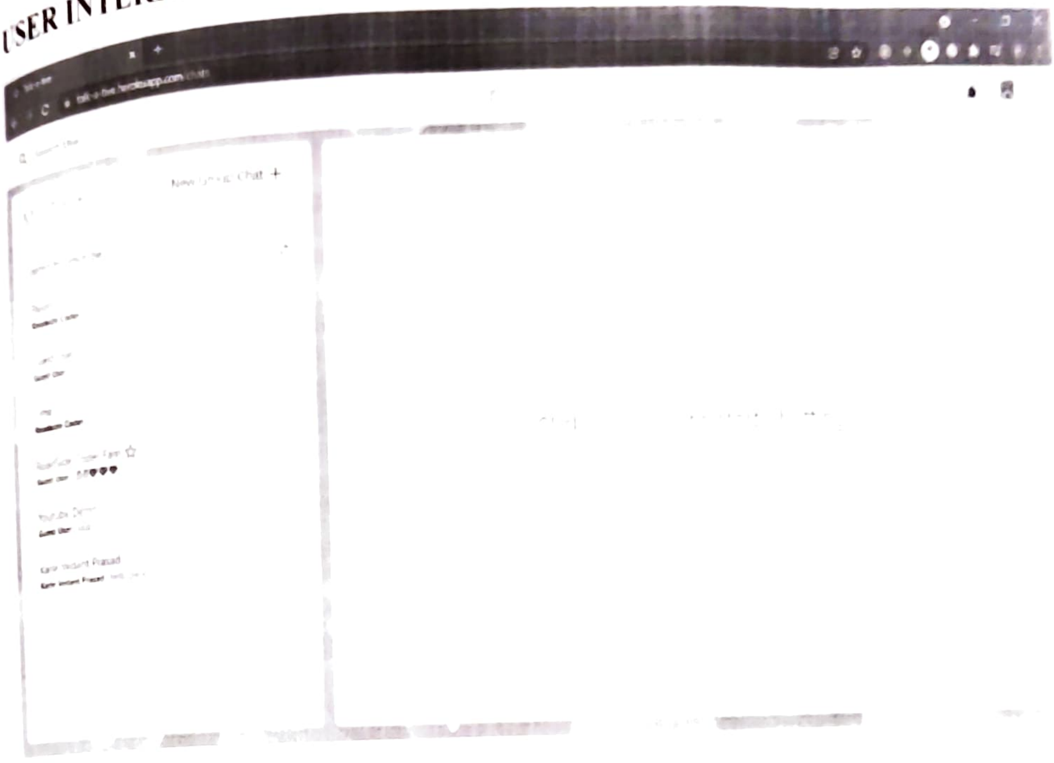
SIGNUP



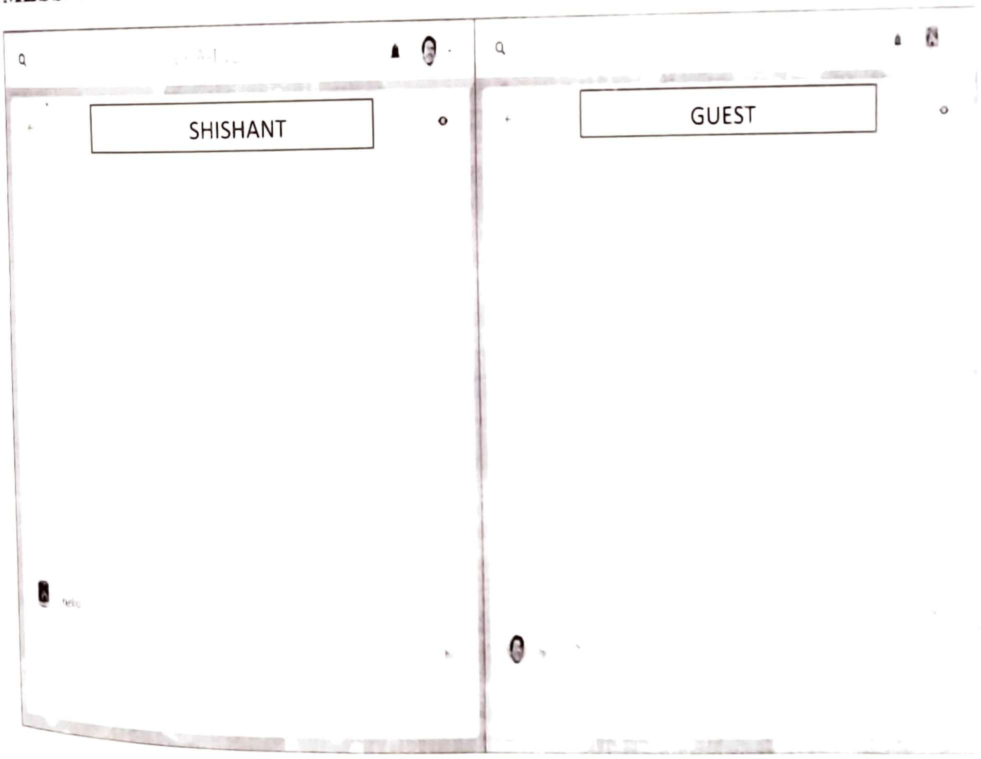
LOGIN



USER INTERFERENCE



MESSAGE INTERFERENCE



5 CONCLUSION

React JS is one of the most famous and incredible front-end innovations in the world today. It offers magnificent execution in any application. Furthermore, React JS offers great similarity in various stages, programs, and gadgets. Subsequent to fostering the Chat App, obviously React JS is simple to learn and helpful to execute. I conclude by saying that the proposed application is a basic task; however, it needs improvement later on the off chance that the engineer has a chance to invest energy into running the tests. It very well may be construed that the talk application created utilizing Node.js, React and in-memory is quicker progressively, with a speed under a second looked at in the application created utilizing JAVA script and Mongodb.

6 Future Plans

We want to add more features such as Whatsapp, Telegram, etc. We want to add videocall and voice call features. We want to add a chat filtration facility to our project, by which we can filter the chats of a particular person. Our chat app is unique when we add chat filtration because popular chatting apps such as Whatsapp. and Telegram do not have chat filtration option. This post is intended to give a stop to their pursuit, as I take care of the multitude of required key parts of how to make an ongoing informing application for portable and web applications with highlights and functionalities.

7 REFRENCES

1.Kiessling, M. The Node Beginner Book; Lulu Press: Morrisville, NC, USA, 2012.

2. Teixeira, P. Professional Node. js: Building Javascript Based Scalable Software; John Wiley & Sons: Hoboken, NJ, USA, 2012.

3. Sidik, B.; Pohan, H.I. Pemrograman Web Dengan HTML. 2nd ed.; Informatika: Bandung, Indonesia, 2010. 4. Purnomosidi, B. Penbangan Sistem Informasi Penegelolaan Inventaris Barang Divisi Pusat Berbasis Web; Bandung: Politeknik-Telkom: Bandung, Indonesia, 2013.

5. Heitkötter, H.; Majchrzak, T.A.; Ruland, B.; Weber, T. Evaluating Frameworks for Creating Mobile Web Apps. In Proceedings of the 9th International Conference on Web Information Systems and Technologies (WEBIST 2013), Aachen, Germany, 8–10 May 2013; pp. 209–221.

6. Brodsky, I. Wireless Computing: A Manager's Guide to Wireless Networking; Van Nostrand Reinhold Company: New York, NY, USA, 1997.

7. Rosa, A.S.; Shalahuddin, M. Pemrograman J2ME Belajar Cepat Pemrograman Perangkat Telekomunikasi Mobile; Informatika Publishing: Bandung, Indonesia, 2008.

8. Goel, A.K.; Pengoria, A.; Kumar, A.; Agrawal, K.K. Composite Movie Recommendation System. In Proceedings of the 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS 2022), Coimbatore, India, 25–26 March 2022; pp. 1273–1278.