Rajiv Gandhi University of Knowledge Technologies

(Catering the Educational Needs of Gifted Rural Youth of A.P)

R.K Valley , Y.S.R Kadapa(Dist)-516330

Project Report

On

We-Connect-U

Submitted by

KUNCHALA SREEKANTH

R170791-E4 CSE

Under the guidance of

B. LINGAMURTHY

Software Engineer,

RK Valley

Rajiv Gandhi University of KnowledgeTechonologies

RK Valley, Kadapa(Dist), Andhra Pradesh, 516330

CERTIFICATE

This is to certify that the project titled "**We-Connect-U**" is a bonafied project submitted by **KUNCHALA SREEKANTH** with ID **R170791** in the department of **COMPUTER SCIENCE AND ENGINEERING** in partial fullfillment of requirement for the award of degree **BACHELOR OF TECHONOLOGY** for the year **2022-2023** carried out the work under the supervision.

N. SATYANANDARAM
(HOD CSE)

B.LINGAMURTHY
(Project Guide)

RAJIV GANDHI UNIVERSITY OF KNOWLEDGE TECHNOLOGIES

(A.P.Government Act 18 of 2008) RGUKT-RK Valley

Vempalli,Kadapa,Andhrapradesh-516330.

DECLARATION

I am, Kunchala Sreekanth (R170791) hereby declare that the project report entitled "We-Connect-U – a web based chatting application using MERN stack" done by is under guidance of

Mr. B.Lingamurthy is submitted in partial fulfilment for the degree of Bachelor of Technology in Computer Science and Engineering during the academic session September 2023 – February 2023 at RGUKT-RK Valley. We also declare that this project is a result of our own effort and has not been copied or imitated from any source. Citations from any websites are mentioned in the references. To the best of my knowledge, the results embodied in this dissertation work have not been submitted to any university or institute for the award of any degree or diploma.

K. Sreekanth(R170791)

ACKNOWLEDGEMENT

The satisfaction that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose constant guidance and encouragement crown all the efforts success.

I am extremely grateful to our respected Director,

Prof. K SANDHYA RANI for keeping excellent academic climate in our institute.

I also express my sincere gratitude to our respected HOD **Mr. N SATYANANDARAM** for his encouragement, overall guidance in viewing this project as a good asset and efforts in bringing out this project.

I would like to convey my special thanks to our guide

Mr. B LINGAMURTHY for guidance, encouragement, co-operation and kindness during the entire duration of course and academics.

Finally, my thanks to all the members who helped me directly and indirectly in the completion of project work. I express my profound gratitude to all my friends and family members for their encouragement.

Abstract :

We Connect U is a web based chatting application is a place where a new user can sign up, if he/she is a existed user can login with the credentials and can chat with contacts, send the attachments. It has become difficult deleting the chat history if there is any private chatting as a subset of that chatting and, we lose many memories if we delete the whole data. It disturbs consistency if we use disappearing mode, also tedious work to delete each message. In order to fulfill the drawback of existed chat applications reading consistency we add a feature which does not disturb main chat on doing private chats and that is the feature of private chat tab where one can enter private chat tab from inside of main tab and we can chat and chatting data will not be stored in database hence it offers full safety.

Table of Contents:

Title	Page No
1 INTRODUCTION	
1.1 Purpose	7
1.2 Document Conventions	7
1.3 Scope	8
1.4 Definition, Acronyms, Abbreviations	8
1.5 Intended Audience and Reading Suggestions	8
2 OVERALL DESCRIPTION	
2.1 Product Perspective	8
2.2 Product Functions	8
2.3 Operating Environment	8
2.4 Assumptions & Dependencies	9
2.5 Software Requirements	9
2.6 Hardware Requirements	9
3 SOFTWARE REQUIREMENT SPECIFICATION	
3.1 Functional Requirements	9
3.2 Non-Functional Requirements	10
3.3 Methodology	10
3.4 Software Requirement Analysis	10
4 SYSTEM DESIGN	
4.1 DFD Diagram	11-12
4.2 ER Diagram	13
4.3 UML Diagrams	14-16
5. INTERFACE	17-24
6. OUTPUT	25-28

1. INTRODUCTION

"We-Connect-U" is a web based chatting application helps to chat with a individual who registered for the application either by searching user name or choosing user from the displayed list of registered members.

It has a login field if the user is already existed user he/she can login with username and password. If the user is new user he/she can register for the application by giving the details like First name, Last name, Email, Password etc.,

By choosing the user as a fore mentioned one can chat and send attachments with other users. Data is stored in the Mongo DB database. The messages are displayed concurrently using the socket.io.

Finally, user can logout finishing the chatting.

1.1 Purpose

The purpose of this project is to chat through web based with other users registered with web-application allow private chatting and to share media between two users.

1.2 Document Conventions

Font	Style	Size
Heading 1 (Cambria)	Heading **(Bold)**	Heading(16)
Subheading(Cambria)	Subheading (Bold)	Subheading(13)
Others (Calibri) Body	Others (Normal)	Others(13)

1.3 Scope

The web-chat application contains the major scope of

1) Users can chat much fast as it developed using the React JS.

2) At a time two users can chat and send attachments.

3) Users need to enter his/her credentials to login to web-application.

1.4 Definition, Acronyms, Abbreviations

React JS	: It is a JavaScript library with declarative features
CSS	: Cascading Style Sheets
CFD	: Context Flow Diagram
ER	: Entity Relationship
SRS	: Software Requirement Specification

2. OVERALL DESCRIPTION

2.1 Product Perspective

We based chatting application is being developed to chat through web with the registered users according to their interests.

2.2 Product Functions

- Shows registered users
- Can chat with any registered user based on individual's choice
- Can send attachments

2.3 Operating Environment

- All operating systems
- All browsers

2.4 Assumptions & Dependencies

One assumption is that the web application is used on a computer with enough performance ability, and the use of an up-to-date internet browser.

2.5 Software Requirements

Scripting languages

1) CSS

2) React JS

2.6 Hardware Requirements

A computer system / laptop with basic configuration.

3. SOFTWARE REQUIREMENT SPECIFICATION

Software Requirement Specification is a description of full software system requirements. Software Requirement Specification describes the behaviour of software from user's point of view. The specified requirements are shown in following table based on module. The functional requirements and non-functional requirements are following:

3.1 Non-Functional Requirements

a) Availability:

The website is available for 24 hours of a day. It is always available for users to contact via email or phone number.

b) Usability:

It has a good user interface. It is user friendly. So, user feel easy to use.

c) Efficiency:

It is efficient for all users. Because it is easy to use and easy to understand. It has simple way of work that use want to do.

3.2 Methodology

This project will follow Incremental Model. This model is splitinto several iterations. New software modules re added in each iteration with no or little change in earlier added modules. The development process can go either sequentially or parallel.

3.3 Software Requirement Analysis

Software requirement analysis is important part of our project. If requirement of project is clear then a project can be done easily. Our objectives for software requirement are:

3.4 Objective/Goals

- User can report any complaint through email or phone number.
- Admin can know everything.

Design Introduction :

Design is the first step in the development phase for any techniques and principles for the purpose of defining a device, a process or system in sufficient detail to permit its physical realization. Once the software requirements have been analysed and specified the software design involves three technical activities – design, coding, implementation and testing that are required to build and verify the software.

DFS diagrams:

DFD Level-0 is also called a Context diagram. It's a basic overview of the whole system or process being analysed or modelled. It's designed to be an at-a-glance view showing the system as a single high level process. With its relationship to external entities.

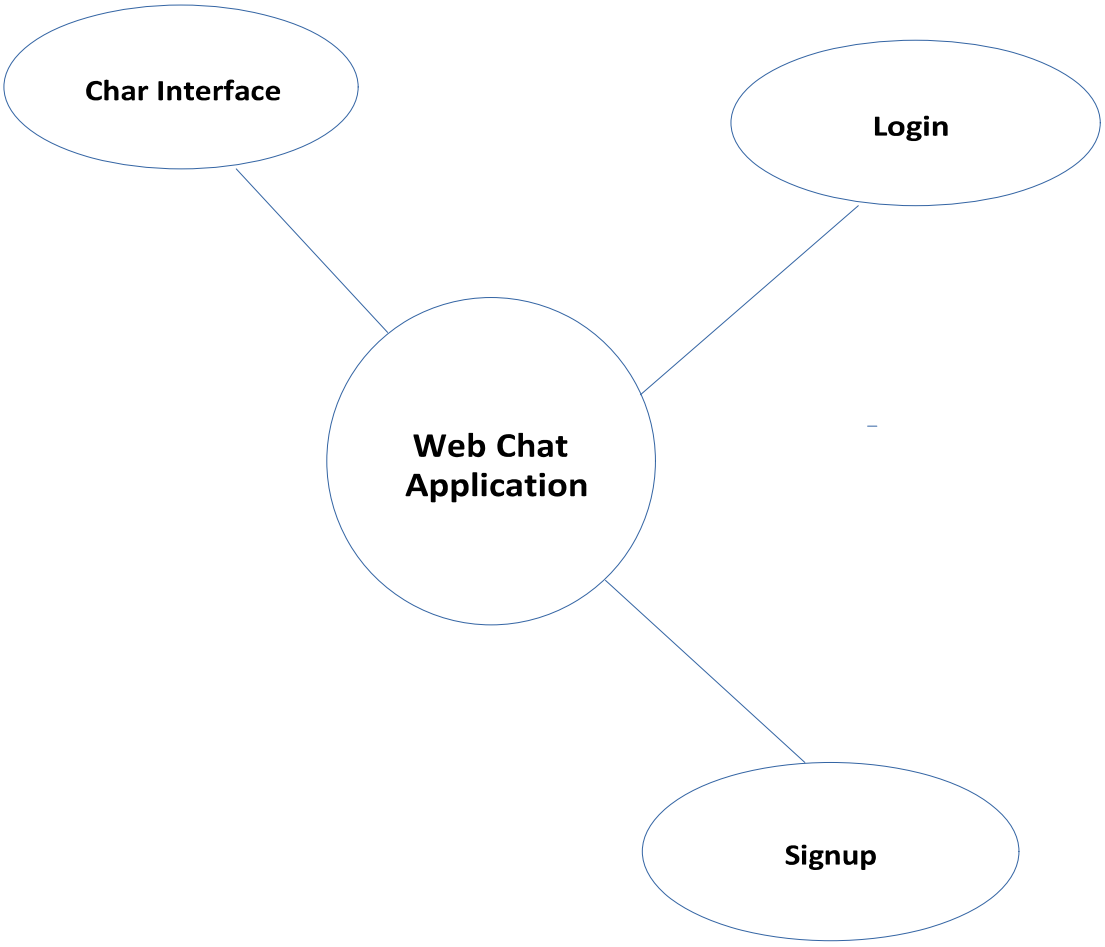
1-Level DFD:

In 1-Level DFD, the context diagram is decomposed into multiple bubbles/processes. In this level, we highlight the main functions of the system and breakdown the high-level process of 0-level DFD into sub processes.

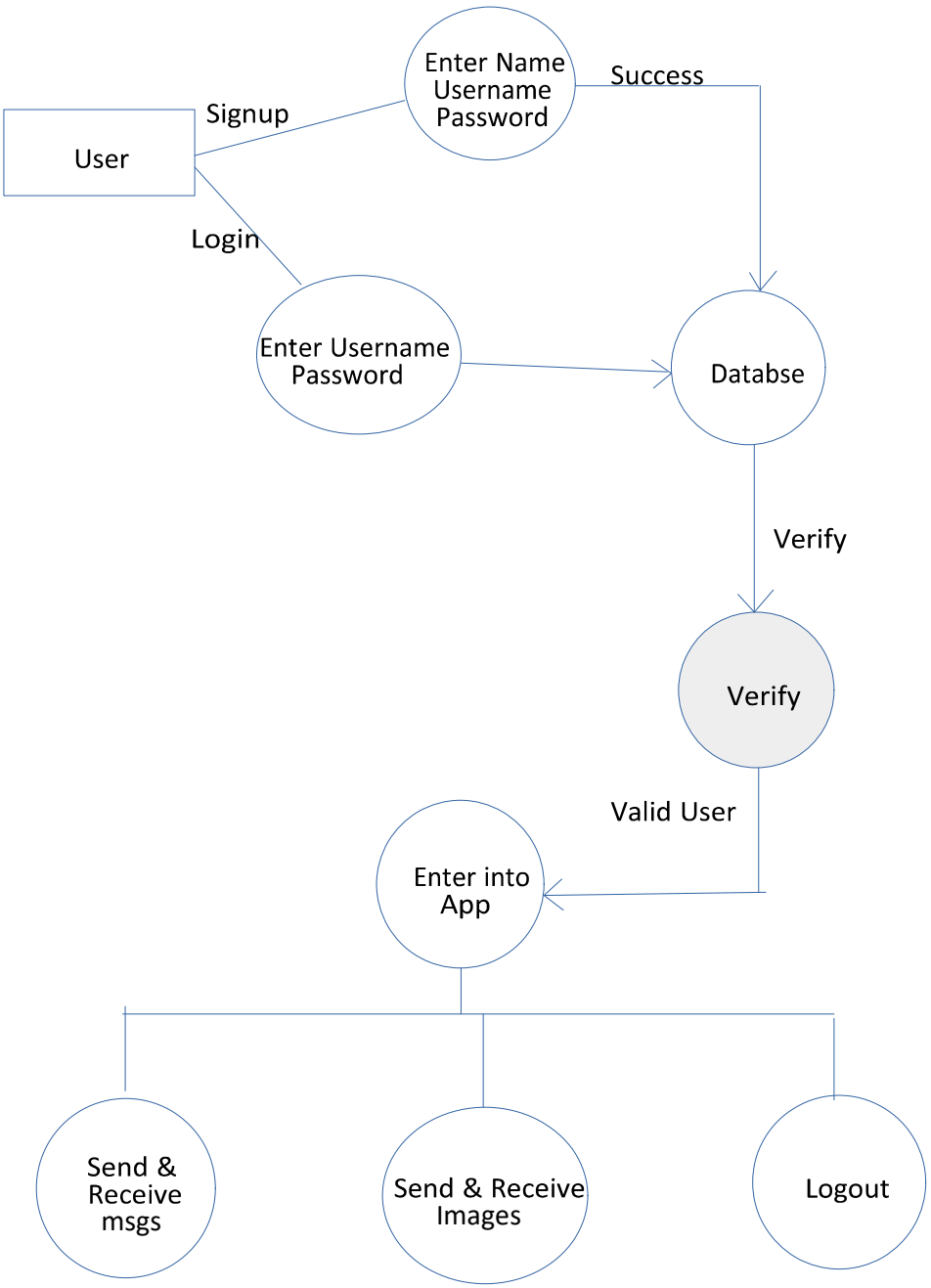
4 SYSTEM DESIGN

4.1 DFD Diagram

Zero Level DFD

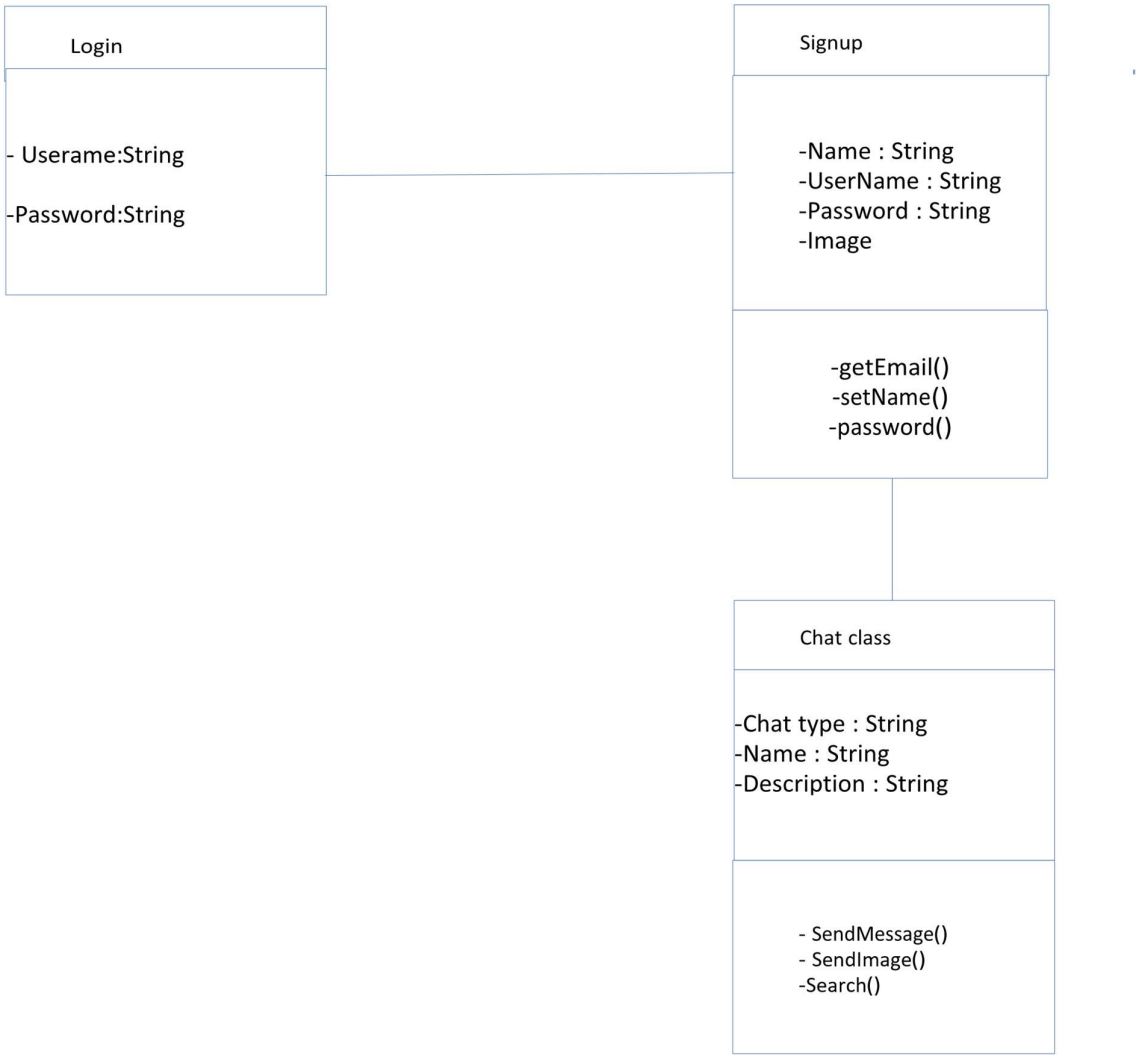


First Level DFD

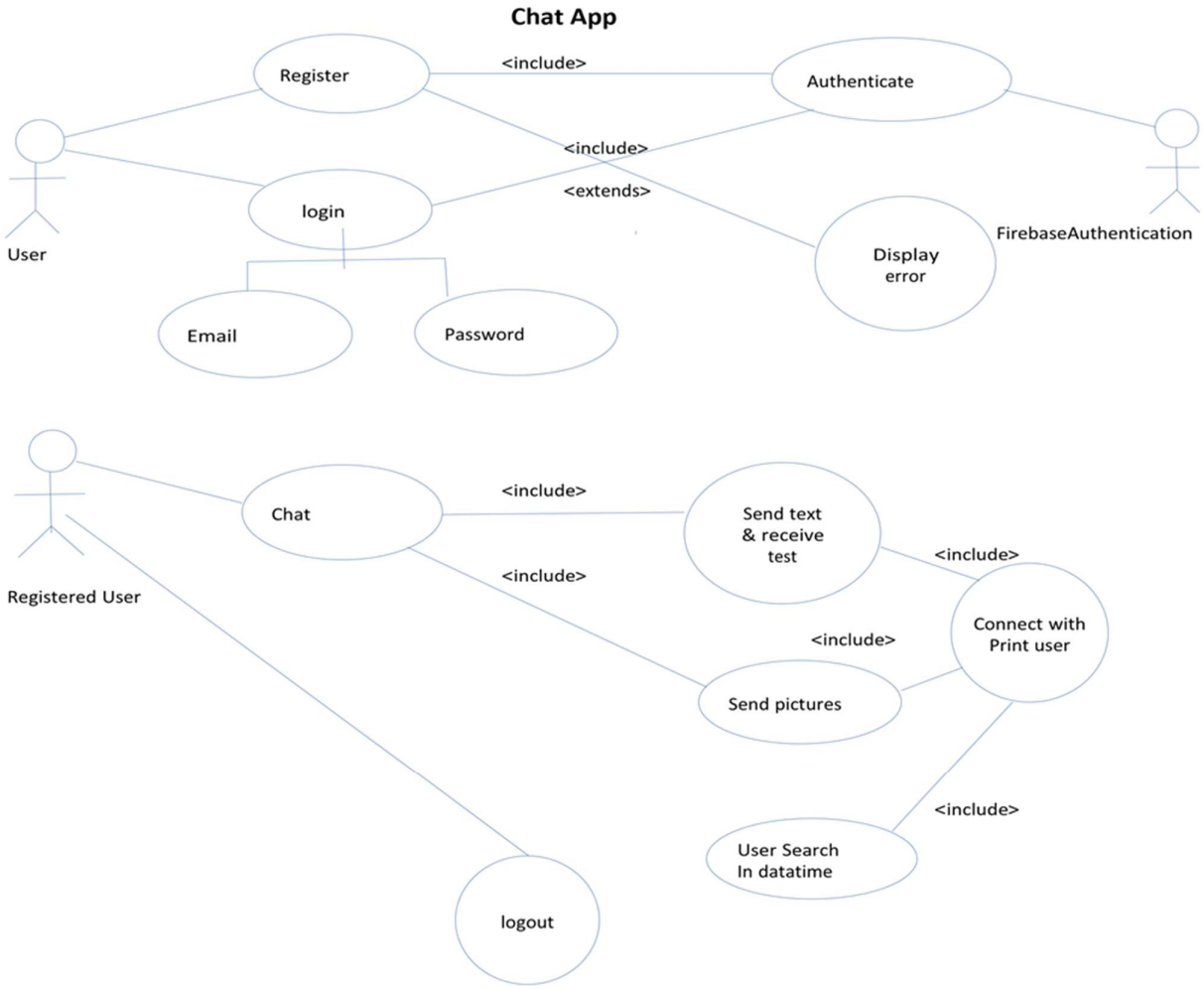


4.2. UML Diagrams

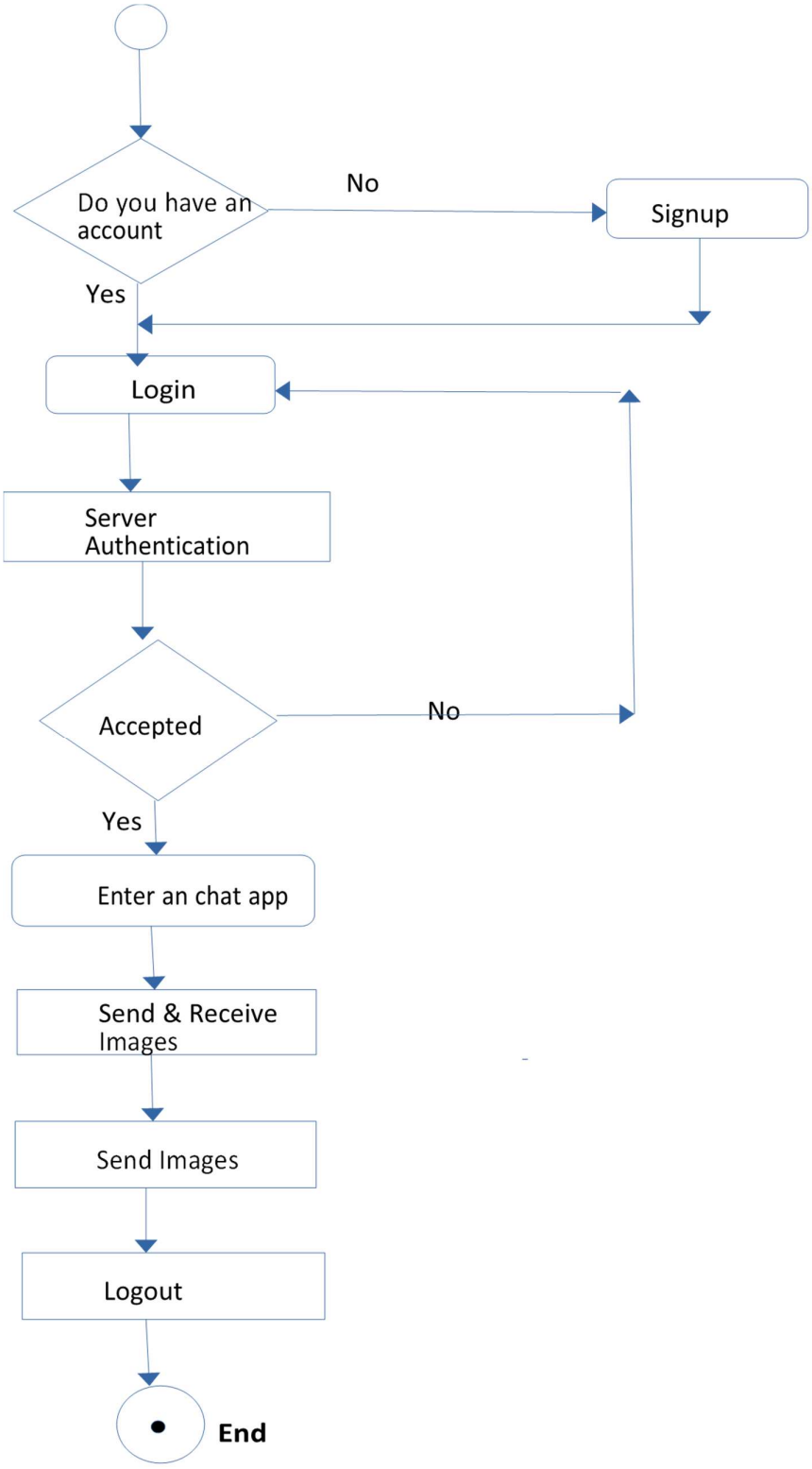
4.2.1 Class Diagrams



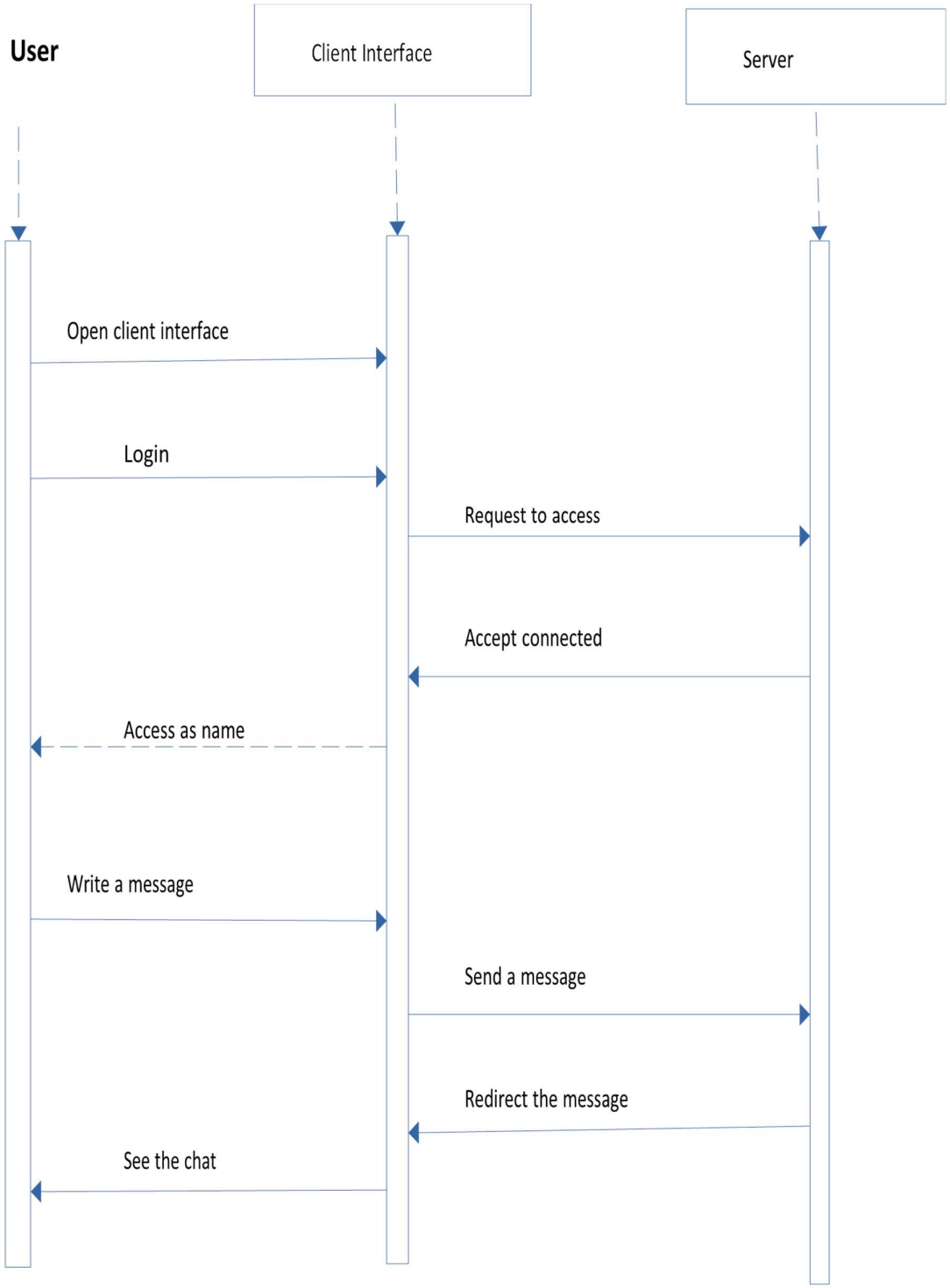
4.3.1 Use Case Diagrams



4.3.3 Activity Diagram



4.3.4 Sequence Diagram



Implementation and System Testing

After all phases have been perfectly done, the system will be implemented to the server and the system can be used.

System Testing

The goal of the system testing process was to determine all faults in our project. The program was subjected to a set of test inputs and many explanations were made and based on these explanations it will be decided whether the program behaves as expected or not. Our project went through two levels of testing. They are:

1. Unit Testing
2. Integration Testing

Unit Testing

Unit testing is commenced when a unit has been created and effectively reviewed. In order to test a single module we need to provide a complete environment i.e. besides the section we would require. The procedures belonging to other units that the unit under test calls non local data structures that module accesses. A procedure to call the functions of the unit test with appropriate parameters.

1. Test for the admin module

Testing admin login form-This form is used for login into administrator of the system. In this form we enter the username and password if both are correct administration page will open otherwise if any of data is wrong it will get redirected back to the login page and again ask the details.

Report Generation: admin can generate report from the main database.

Integration Testing

In the integration testing we test various combination of project module by providing the input. The primary objective is to test the module interfaces in order to confirm that no errors are occurring when one module invokes the other module.

Sample Code:

Register File:

```
import React from 'react'
import { VStack } from '@chakra-ui/layout'
import { FormControl,FormLabel } from '@chakra-ui/react'
import { Input,InputGroup, InputRightElement } from "@chakra-ui/input"
import react, { useState } from "react";
import { Button } from "@chakra-ui/button";
import { useToast } from '@chakra-ui/react'
import axios from 'axios';
import { useHistory } from 'react-router-dom';


const SignUp = () => {
    const [show, setShow] = useState(false);
    const [pic, setPic] = useState();
    const [picLoading, setPicLoading] = useState(false);
    const toast = useToast();
    const history = useHistory();
    const handleClick = () => setShow(!show);
    const submitHandler = async () => {
        setPicLoading(true);
        if (!name || !email || !password || !confirmpassword) {
          toast({
            title: "Please Fill all the Feilds",
            status: "warning",
            duration: 5000,
            isClosable: true,
            position: "bottom",
          });
          setPicLoading(false);
          return;
        }
        if (password !== confirmpassword) {
          toast({
            title: "Passwords Do Not Match",
            status: "warning",
```

```
            duration: 5000,
            isClosable: true,
            position: "bottom",
});
          return;
        }
        console.log(name, email, password, pic);
        try {
          const config = {
            headers: {
              "Content-type": "application/json",
            },
          };
          const { data } = await axios.post(
            "/api/user",
            {
              name,
              email,
              password,
              pic,
            },
            config
          );
          console.log(data);
          localStorage.setItem("userInfo", JSON.stringify(data));
          setPicLoading(false);
          history.push("/chats");
        } catch (error) {
          toast({
            title: "Error Occured!",
            description: error.response.data.message,
            status: "error",
            duration: 5000,
            isClosable: true,
            position: "bottom",
          });
          setPicLoading(false);
        }
      };

}

export default SignUp
```

Login File:

```
import { Button } from "@chakra-ui/button";
import { FormControl, FormLabel } from "@chakra-ui/form-control";
import { Input, InputGroup, InputRightElement } from "@chakra-ui/input";
import { VStack } from "@chakra-ui/layout";
import { useState } from "react";
import axios from "axios";
import { useToast } from "@chakra-ui/react";
import { useHistory } from "react-router-dom";

const Login = () => {
  const submitHandler = async () => {
    setLoading(true);
    if (!email || !password) {
      toast({
        title: "Please Fill all the Feilds",
        status: "warning",
        duration: 5000,
        isClosable: true,
        position: "bottom",
      });
      setLoading(false);
      return;
    }

    // console.log(email, password);
    try {
      const config = {
        headers: {
          "Content-type": "application/json",
        },
      };

      const { data } = await axios.post(
        "/api/user/login",
        { email, password },
        config
      );

      // console.log(JSON.stringify(data));
      toast({
        title: "Login Successful",
        status: "success",
        duration: 5000,
        isClosable: true,
        position: "bottom",
      });
      localStorage.setItem("userInfo", JSON.stringify(data));
```

```
      setLoading(false);
      history.push("/chats");
    } catch (error) {
      toast({
        title: "Error Occured!",
        description: error.response.data.message,
        status: "error",
        duration: 5000,
        isClosable: true,
        position: "bottom",
      });
      setLoading(false);
    }
  };



  );
};

export default Login;
```

Group Chat:

```
const GroupChatModal = ({children}) => {

    const toast = useToast();
    const { user, chats, setChats } = ChatState();

    const handleGroup = (userToAdd) => {
        if (selectedUsers.includes(userToAdd)) {
          toast({
            title: "User already added",
            status: "warning",
            duration: 5000,
            isClosable: true,
            position: "top",
          });
          return;
        }
        setSelectedUsers([...selectedUsers, userToAdd]);
      };


    const handleSearch = async (query) => {
        setSearch(query);
        if (!query) {
          return;
        }
```

```
        try {
          setLoading(true);
          const config = {
            headers: {
              Authorization: `Bearer ${user.token}`,
            },
          };
          const { data } = await axios.get(`/api/user?search=${search}`,
config);

          setLoading(false);
          setSearchResult(data);
        } catch (error) {
          toast({
            title: "Error Occured!",
            description: "Failed to Load the Search Results",
            status: "error",
            duration: 5000,
            isClosable: true,
            position: "bottom-left",
          });
        }
      };
        try {
          const config = {
            headers: {
              Authorization: `Bearer ${user.token}`,
            },
          };
          const { data } = await axios.post(
            `/api/chat/group`,
            {
              name: groupChatName,
              users: JSON.stringify(selectedUsers.map((u) => u._id)),
            },
            config
          );
          setChats([data, ...chats]);
          onClose();
          toast({
            title: "New Group Chat Created!",
            status: "success",
            duration: 5000,
            isClosable: true,
            position: "bottom",
          });
        } catch (error) {
```

```
      toast({
        title: "Failed to Create the Chat!",
        description: error.response.data,
        status: "error",
        duration: 5000,
        isClosable: true,
        position: "bottom",
      });
    }
  };

    return (
      <>
        <FormControl>
          <Input
            placeholder="Add Users eg: hari, lithu, mouni.."
            mb={1}
            onChange={(e) => handleSearch(e.target.value)}
          />
        </FormControl>

        <Box w="100%" d="flex" flexWrap="wrap">
          {selectedUsers.map((u) => (
            <UserBadgeItem
              key={u._id}
              user={u}
              handleFunction={() => handleDelete(u)}
            />
          ))}
        </Box>


        {loading ? (
          // <ChatLoading />
          <div>Loading...</div>
        ) : (
          searchResult
            ?.slice(0, 4)
            .map((user) => (
              <UserListItem
                key={user._id}
                user={user}
                handleFunction={() => handleGroup(user)}
              />
            ))
        )}


            </ModalBody>
```

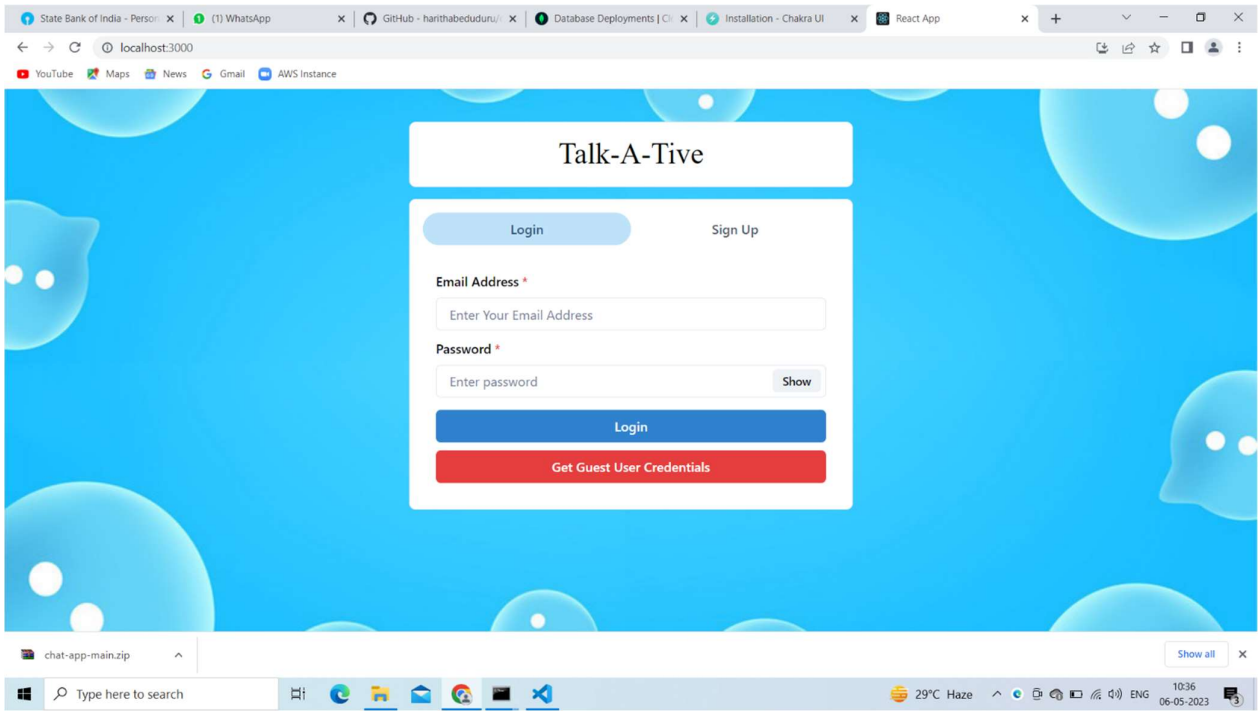
```
                <ModalFooter>
                  <Button colorScheme='blue' onClick={handleSubmit}>
                    Create Chat
                  </Button>

                </ModalFooter>
              </ModalContent>
            </Modal>
          </>
        )
      }
```

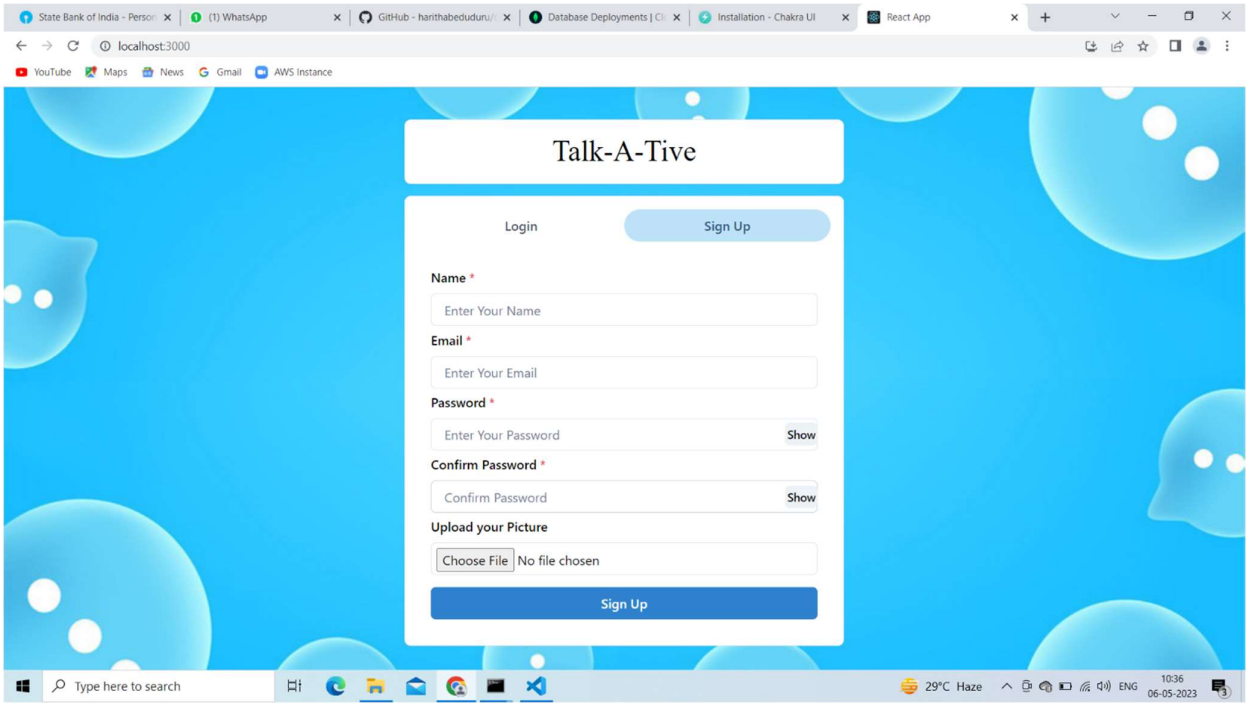
Notifications:

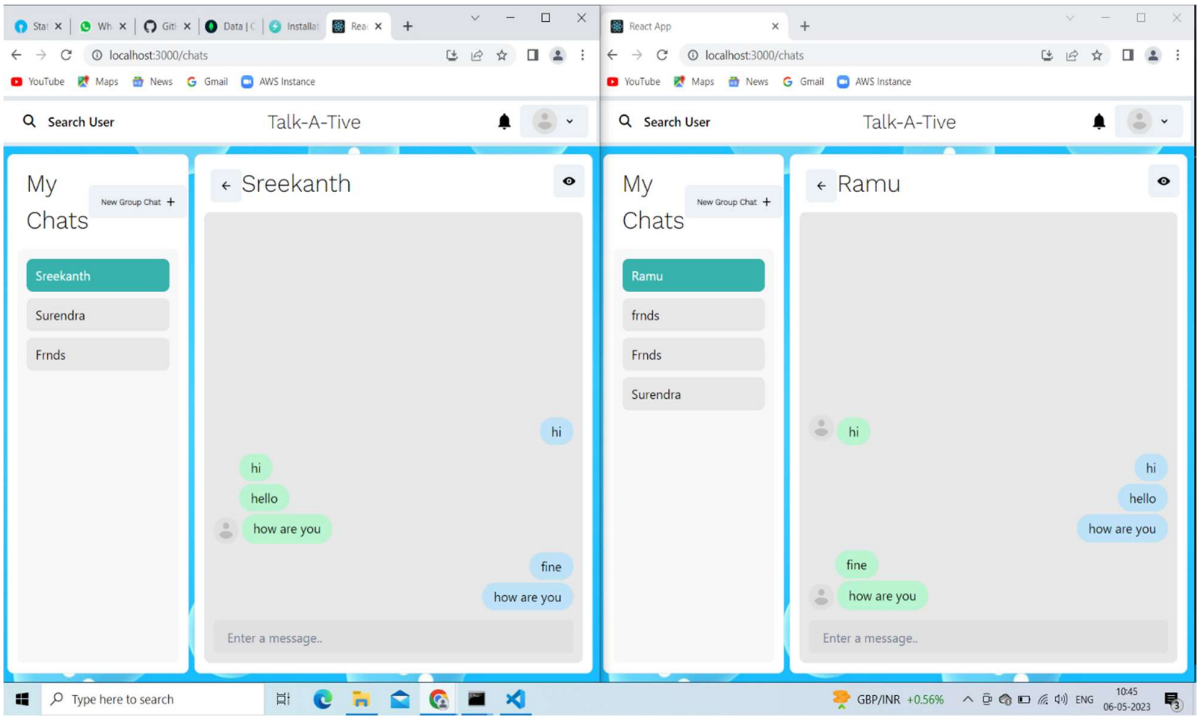
```
import React, { useState} from 'react';
import { Box,Text } from '@chakra-ui/layout';
const SideDrawer = () => {
    const logoutHandler = () => {
        localStorage.removeItem('userInfo');
        history.push('/');
    }
return <>
            <div>
                <Menu>
                    <MenuButton p={1}>
                        <NotificationBadge
                            count = {notification.length}
                            effect={Effect.SCALE}
                        />
                    <BellIcon fontSize='2xl' m={1}/>
                    </MenuButton>
                        <MenuList pl={2}>
                            {!notification.length && "No New Messages"}
                            {notification.map((notif) => (
                            <MenuItem
                                key={notif._id}
                                onClick={() => {
                                    setSelectedChat(notif.chat);
                                    setNotification(notification.filter((n)=>
n !== notif));

                                }}
                            > {notif.chat.isGroupChat
                                ? `New Message in ${notif.chat.chatName}`
                                : `New Message from ${getSender(user,
notif.chat.users)}`}
                            </MenuItem>
```

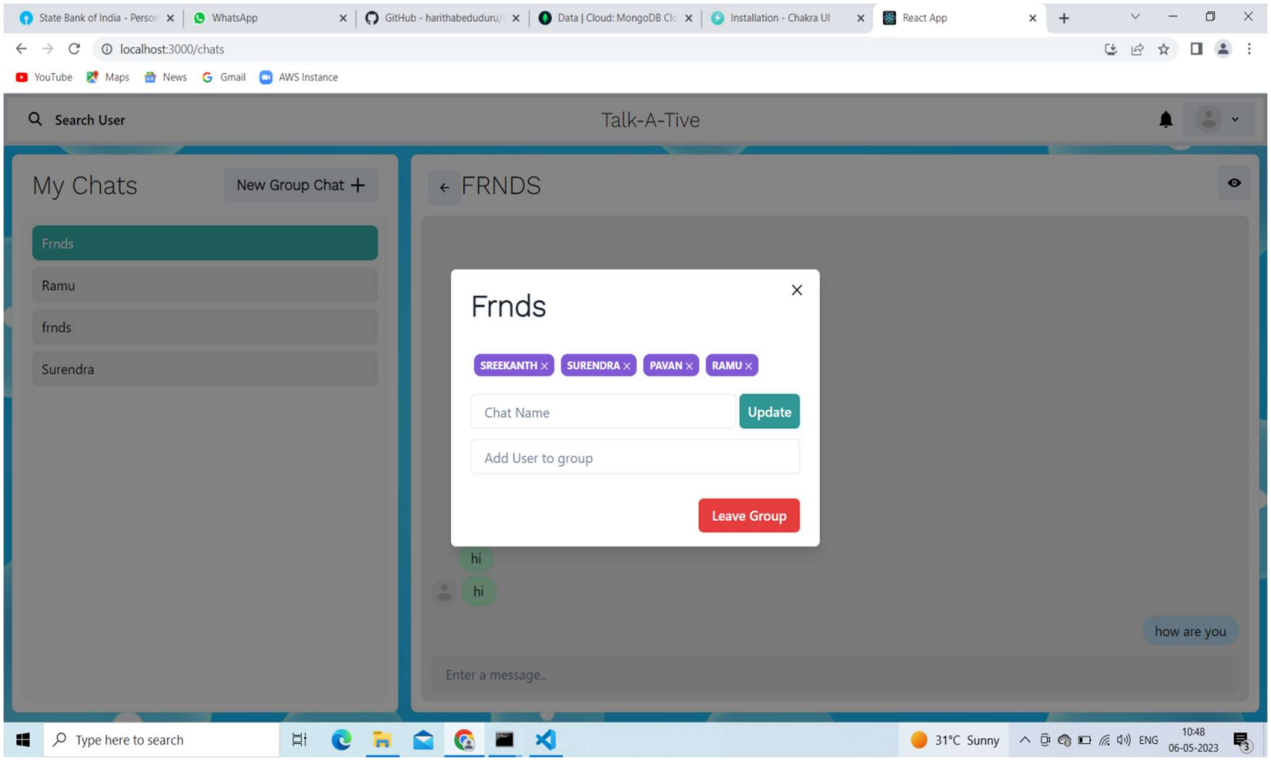
```
                        ))}
                    </MenuList>

                </Menu>
            </div>
        </Box>


        <Drawer placement='left' onClose={onClose} isOpen={isOpen}>
            <DrawerOverlay />
        </Drawer>
    </>
}
export default SideDrawer;
```

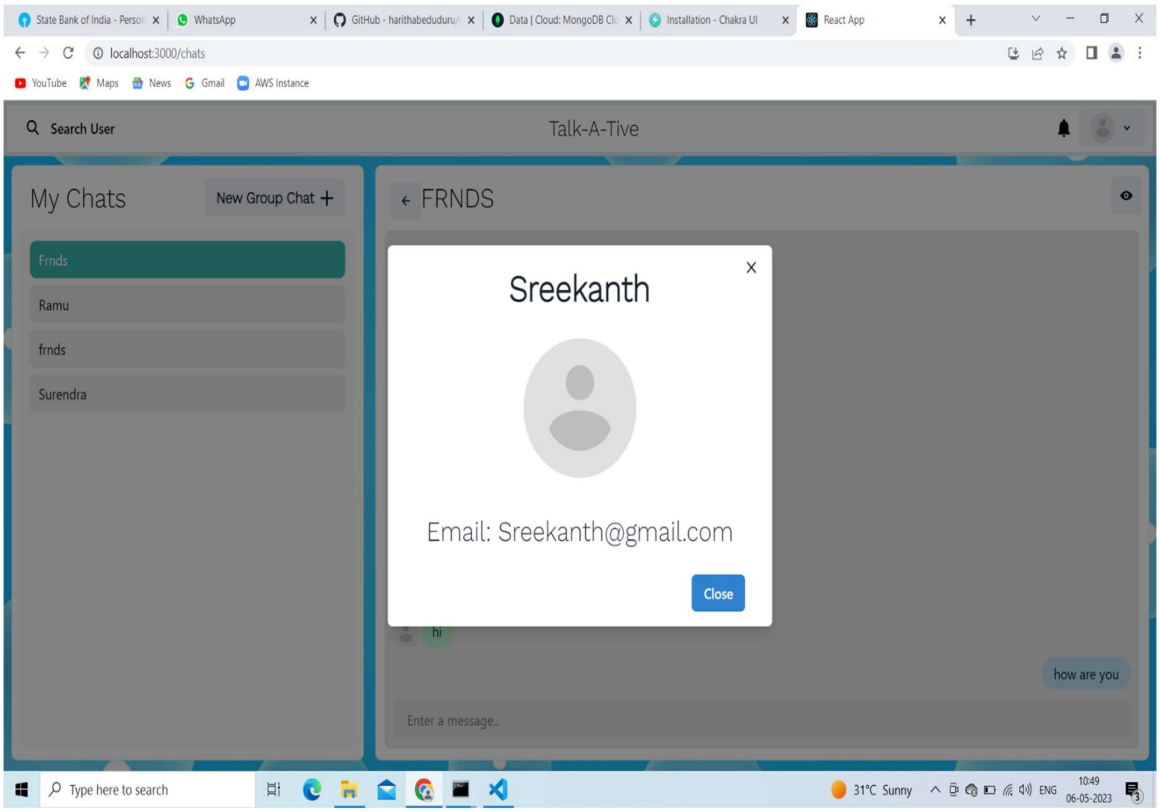
Output:

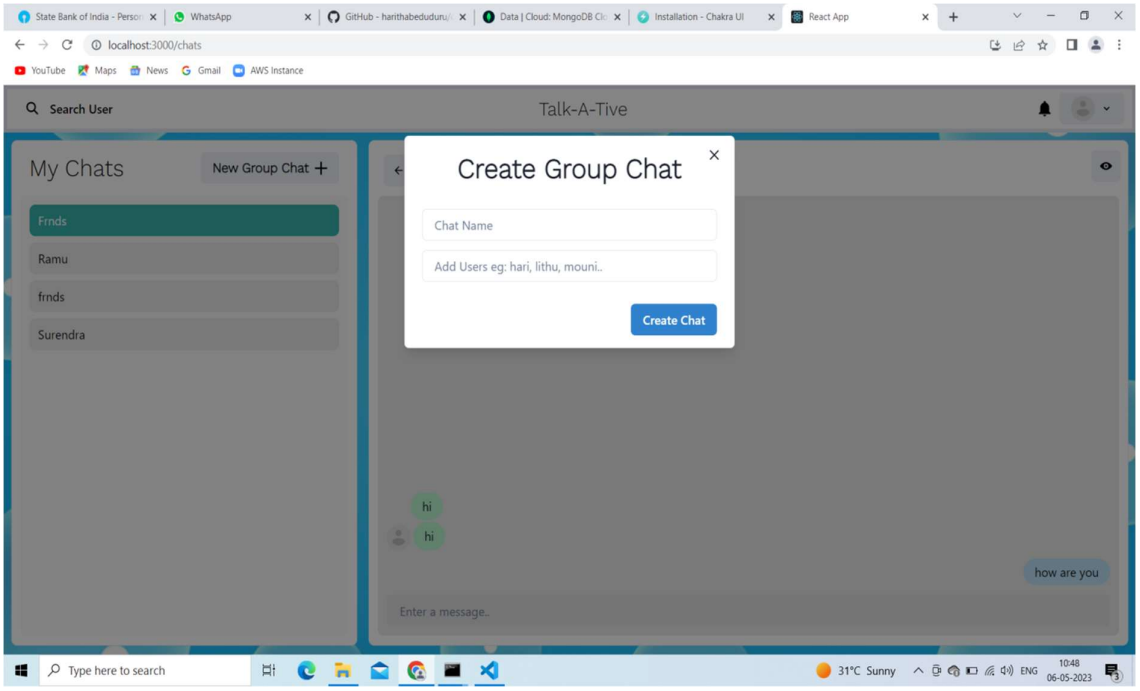



Talk-A-Tive
Login
Sign Up
Name *
Enter Your Name
Email *
Enter Your Email
Password *
Enter Your Password
Show
Confirm Password *
Confirm Password
Show
Upload your Picture
Choose File
No file chosen
Sign Up


Search User
Talk-A-Tive
My Chats
New Group Chat
Sreekanth
Surendra
Frnds
Sreekanth
hi
hi
hello
how are you
fine
how are you
Enter a message..
Search User
Talk-A-Tive
My Chats
New Group Chat
Ramu
frnds
Frnds
Surendra
Ramu
hi
hi
hello
how are you
fine
how are you
Enter a message..


localhost:3000/chats
Search User
Talk-A-Tive
My Chats
New Group Chat +
Frnds
Ramu
frnds
Surendra
FRNDS
Frnds
SREEKANTH
SURENDRA
PAVAN
RAMU
Chat Name
Update
Add User to group
Leave Group
hi
hi
how are you
Enter a message..
Type here to search
31°C Sunny
ENG
10:48
06-05-2023


localhost:3000/chats
Search User
Talk-A-Tive
My Chats
New Group Chat +
Frnds
Ramu
frnds
Surendra
FRNDS
Sreekanth
Email: Sreekanth@gmail.com
Close
hi
how are you
Enter a message..
Type here to search
31°C Sunny
ENG
10:49
06-05-2023



References:

chakra-ui.com

socket.io

MERN Stack Chat App with Socket.IO Tutorial – YouTube