Full Stack Report
(2020-2021)

UV WhatsApp

Department of Computer Engineering & Applications
Institute of Engineering and Technology

Submitted by:
Vipul (181500801)
Umesh Pratap Singh (181500767)
Harshita Bhardwaj (181500261)

Supervised By:
Dr. Pankaj Kapoor
Technical Trainer
Department of Computer Engineering and Application

Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406

Declaration

We hereby declare that the work which is being presented in the Full Stack II **"**UV WhatsApp**",** in partial full fillment of the requirements for Full Stack viva voce, is an authentic record of our own work carried under the supervision of **Mr. Pankaj Kapoor, Assistant Professor, GLA University, Mathura.**

Harshita Bhardwaj (181500261)
Sign: _____

Vipul (181500801)
Sign: _____

Umesh Pratap Singh (181500767)
Sign: _____

Course: B.Tech(CSE)
Year: 3rd
Semester: VI
Members GitHub Id's:

- https://github.com/vipulgupta22
- https://github.com/harshitabhardwaj16
- https://github.com/thakurups

Department of computer Engineering and Applications
GLA University, Mathura
17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha,
Mathura – 281406

Certificate

This is to certify that the project entitled "UV WhatsApp" carried out in Full Stack II is the work done by Vipul, Harshita Bhardwaj, Umesh Pratap Singh and is submitted in partial full fillment of the requirements for the award of degree Bachelor of Technology (Computer Science and Engineering).

Signature of Supervisor:

Name of Supervisor: Dr. Pankaj Kapoor

Date:

Acknowledgement

It is our pleasure to acknowledge the assistance of a number of people without whose help this project would not have been possible.
First and foremost, We I would like to express our gratitude to **Dr. Pankaj Kapoor** our project mentor, for providing invaluable Encouragement, guidance and assistance. We would like to thank my co-team members for their complete support throughout in finishing the mentioned project accurately. After doing this project We can confidently say that this experience has not only enriched us with technical knowledge but also has unparsed the maturity of thought and vision, the attributes required for being a professional.

Abstract

WhatsApp employs the use of internet to send text messages, multimedia messages, and user location messages to other users by use of regular cellular mobile phones. WhatsApp faced competition from Asian-based messaging services such as WeChat, Viber, LINE among Skype and Telegram. WhatsApp was first founded by Jan Koum and Brian Acton in 2009 former workers of Yahoo! By January 2015, WhatsApp was the most globally popular messaging app that had more than 600 million active users worldwide a number that grew to one billion by February 2016 with over 30 billion messages being sent daily. Since it was first developed, it went through some improvements with the latest ones coming through its ability to share documents and also the encryption feature that came very recently including the improvement that enables it to be used to make voice calls. Its usage also grew steadily, and it is now used by over one billion people around the world with over 30 billion messages being sent daily. WhatsApp uses the internet to send text messages, images, videos, documents, audio, media and user location messages to other users by use of regular cellular mobile phones.

Table of Contents

Chapter-1
Introduction

1.1. Overview

UV-WhatsApp Messenger is a proprietary, cross-platform instant messaging service for web without having to pay for SMS. UV-WhatsApp a dream came true of talking to a friend sitting Ocean Apart through Radio-Waves. There is the involvement of internet by logging and signup in a personal computer to join the chat rooms. It is that where people wanted to stay in touch with their loved ones and their families.

1.2. Motivation

As social media is trending nowadays, there are various social connecting platforms, hence there are lot of chances to improve this connection. In order to improve something, it is important to know what the thing is. Hence creating a Chat App is a initiate.
The goal of the WebApp is to connect people socially.

1.3. Project Plan

1.3.1. Objective

The main objective of this project is to create a React App for Practical implementation of Rect.js. In this Web App the user will be able to sign in with google and can create public groups to chat with other people.

1.3.2. Scope

UV-WhatsApp is a messaging app, with over a billion and a half monthly users. I use it almost every day to keep in touch with teams, friends, and family all over the world. It's simple and effective. It's a no-frills, lean, and reliable chatting app.

1.4. Drawbacks in Existing System

- No Emoji chats in our app.
- There is no privacy of chats easily accessible.
- And No adding status and adding more features.

Chapter-2
Software Requirement Analysis

2.1. Hardware Requirements

- Pentium Processor
- 60 MB of free hard-drive space
- 128 MB of RAM

2.2. Software Requirements

- Operating System: Windows (Vista/7 or above)
- Web Browser: IE 10 or above, Mozilla FF 31 or Google Chrome
- Drivers: Java Runtime Environment
- Tools: GitHub, VSCode
- Front-end: HTML, CSS, JavaScript

2.3. Installation of VS Code

VS Code is a free code editor, which runs on the macOS, Linux, and Windows operating systems.
VS Code is lightweight and should run on most available hardware and platform versions. You can review the System Requirements to check if your computer configuration is supported.

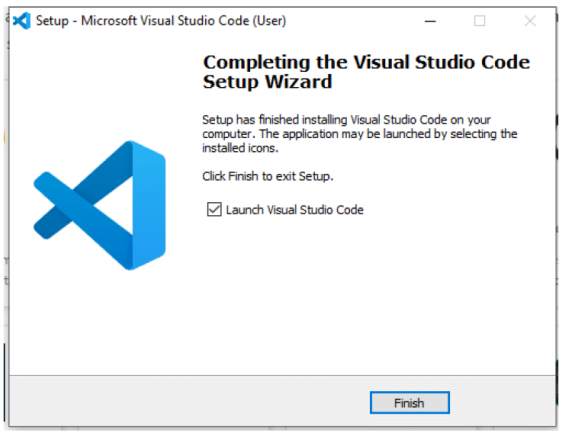


Fig1. Finish up Installing.

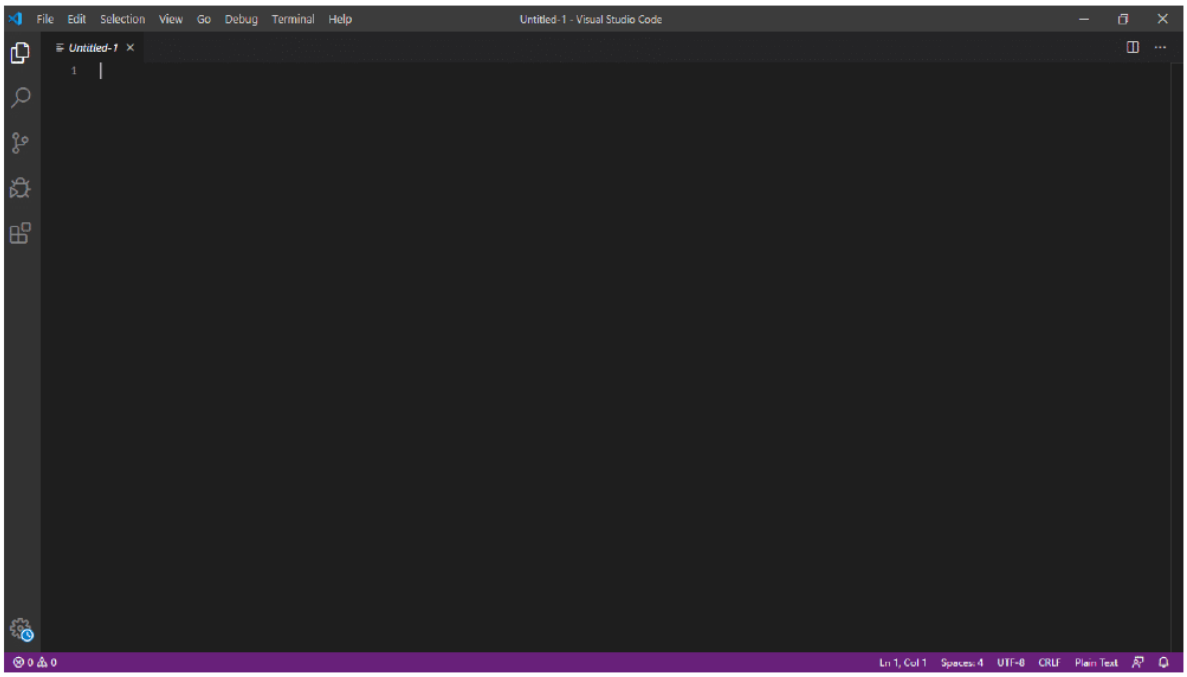


Fig2. VS Code Window.

2.4. Specific Requirements

2.4.1 Languages Used

HTML:

HTML stands for **Hyper Text Mark-up Language**, which is the most widely used language on Web to develop web pages. HTML was created by Berners-Lee in late 1991 but "HTML 2.0" was the first standard HTML specification which was published in 1995. HTML 4.01 was a major version of HTML and it was published in late 1999. Though HTML 4.01 version is widely used but currently we are having HTML-5 version which is an extension to HTML 4.01, and this version was published in 2012.

I will list down some of the key advantages of learning HTML:

• **Create Web site** - You can create a website or customize an existing web template if you know HTML well.

• **Become a web designer** - If you want to start a carrer as a professional web designer, HTML and CSS designing is a must skill.

• **Understand web** - If you want to optimize your website, to boost its speed and performance, it is good to know HTML to yield best results.

• **Learn other languages** - Once you understands the basic of HTML then other related technologies like java script, php, or angular are become easier to understand.

CSS:

Cascading **S**tyle **S**heets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable. CSS handles the look and feel part of

a web page. Using CSS, you can control the colour of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.
CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the mark-up languages HTML or XHTML.
• **CSS saves time** − You can write CSS once and then reuse same sheet in multiple HTML pages. You can define a style for each HTML element and apply it to as many Web pages as you want.
• **Pages load faster** − If you are using CSS, you do not need to write HTML tag attributes every time. Just write one CSS rule of a tag and apply it to all the occurrences of that tag. So less code means faster download times.
• **Easy maintenance** − To make a global change, simply change the style, and all elements in all the web pages will be updated automatically.
• **Superior styles to HTML** − CSS has a much wider array of attributes than HTML, so you can give a far better look to your HTML page in comparison to HTML attributes.
• **Multiple Device Compatibility** − Style sheets allow content to be optimized for more than one type of device. By using the same HTML document, different versions of a website can be presented for handheld devices such as PDAs and cell phones or for printing.

JavaScript:

JavaScript is a lightweight, interpreted programming language. It is designed for creating network-centric applications. It is complimentary to and integrated with Java. JavaScript is very easy to implement because it is integrated with HTML. It is open and cross-platform.

Fig3. HTML vs CSS vs JAVASCRIPT

React:

React is a declarative, efficient, and flexible JavaScript library for building user interfaces. It lets you compose complex UIs from small and isolated pieces of code called “components”. React (also known as React.js or ReactJS) is an open-source, front end, JavaScript library for building user interfaces or UI components. It is maintained by Facebook and a community of individual developers and companies. React can be used as a base in the development of single-page or mobile applications. However, React is only concerned with state management and rendering that state to the DOM, so creating React applications usually requires the use of additional libraries for routing, as well as certain client-side functionality.

Firebase:

Firebase is a platform developed by Google for creating mobile and web applications. It was originally an independent company founded in 2011. In 2014, Google acquired the platform and it is now their flagship offering for app development. Firebase has been claimed to be used by Google to track users without their knowledge. On July 14, 2020, a lawsuit was filed accusing Google of violating federal wire tap law and California privacy law. It stated that through Firebase, Google collected and stored user data, logging what the user was looking at in many types of apps, despite the user following Google's own instructions to turn off the web and app activity collected by the company. Google Firebase is a Google-backed application development software that enables developers to develop iOS, Android and Web apps. Firebase provides tools for tracking analytics, reporting and fixing app crashes, creating marketing and product experiment.

Chapter-3
Software Designs

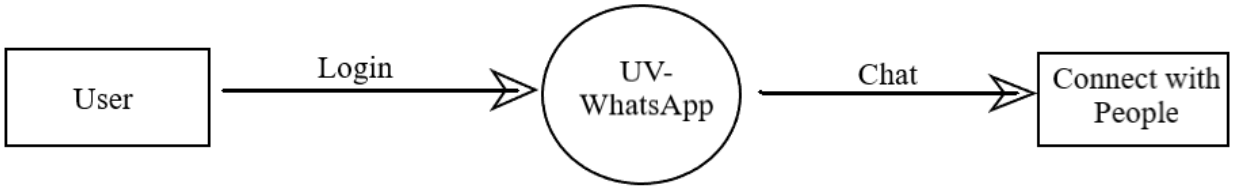
3.1 Use Case diagram

A **use case diagram** is a dynamic or behaviour diagram in UML. Use case diagrams model the functionality of a system using actors and use cases. Use cases are a set of actions, services, and functions that the system needs to perform.

Use Case Diagram

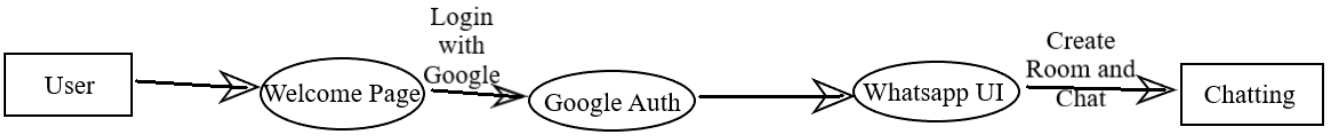
3.2 Data Flow Diagram

Level 0 DFD:



0 Level DFD

Level 1 DFD:



1 Level DFD

Chapter-4
Testing

4.1 Introduction

The implementation phase of software development is concerned with translating design specification into source code. The preliminary goal of implementation is to write source code and internal documentation so that conformance of the code to its specifications can be easily verified, and so that debugging, testing and modifications are eased. This goal can be achieved by making the source code as clear and straightforward as possible. Simplicity, clarity and elegance are the hallmark of good programs, obscurity, cleverness, and complexity are indications of inadequate design and misdirected thinking.
Source code clarity is enhanced by structured coding techniques, by good coding style, by, appropriate supporting documents, by good internal comments, and by feature provided in modern programming languages. The implementation team should be provided with a well-defined set of software requirement, an architectural design specification, and a detailed design description. Each team member must understand the objectives of implementation.

4.2. Error

The term error is used in two ways. It refers to the difference between the actual output of software and the correct output, in this interpretation, error is essential a measure of the difference between actual and ideal. Error is also to used to refer to human action that result in software containing a defect or fault.

4.3. Fault

Fault is a condition that causes to fail in performing its required function. A fault is a basic reason for software malfunction and is synonymous with the commonly used term Bug.

4.4. Failure

Failure is the inability of a system or component to perform a required function according to its specifications. A software failure occurs if the behaviour of the software is the different from the specified behaviour. Failure may be caused due to functional or performance reasons.

a. Unit Testing

The term unit testing comprises the sets of tests performed by an individual programmer prior to integration of the unit into a larger system.
A program unit is usually small enough that the programmer who developed it can test it in great detail, and certainly in greater detail than will be possible when the unit is integrated into an evolving software product. In the unit testing the programs are tested separately, independent of each other. Since the check is done at the program level, it is also called program teasing.

b. Module Testing

A module and encapsulates related component. So can be tested without other system module.

c. Subsystem Testing

Subsystem testing may be independently design and implemented common problems are sub-system interface mistake in this checking we concenton it. There are four categories of tests that a programmer will typically perform on a program unit.

i Functional test
ii Performance test
iii Stress test
iv Structure test

4.5 Functional Test

Functional test cases involve exercising the code with Nominal input values for which expected results are known; as well as boundary values (minimum values, maximum values and values on and just outside the functional boundaries) and special values.

4.6 Performance Test

Performance testing determines the amount of execution time spent in various parts of the unit, program throughput, response time, and device utilization by the program unit. A certain amount of avoid expending too much effort on fine-tuning of a program unit that contributes little to the overall performance of the entire system. Performance testing is most productive at the subsystem and system levels.

4.7 Stress Test

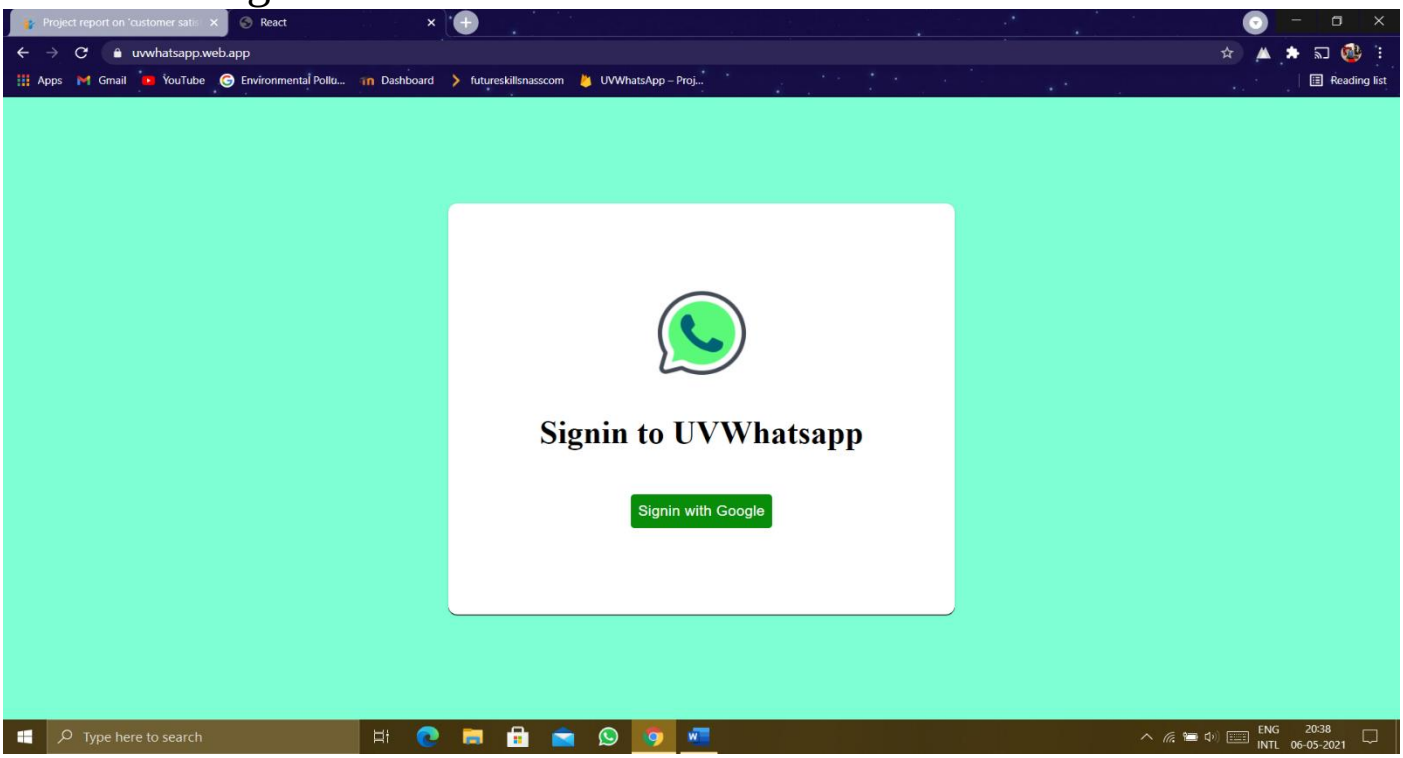
Stress test are those designed to intentionally break the unit. A great deal can be learned about the strengths and limitations of a program by examining the manner in which a program unit breaks.

4.8 Structure Test

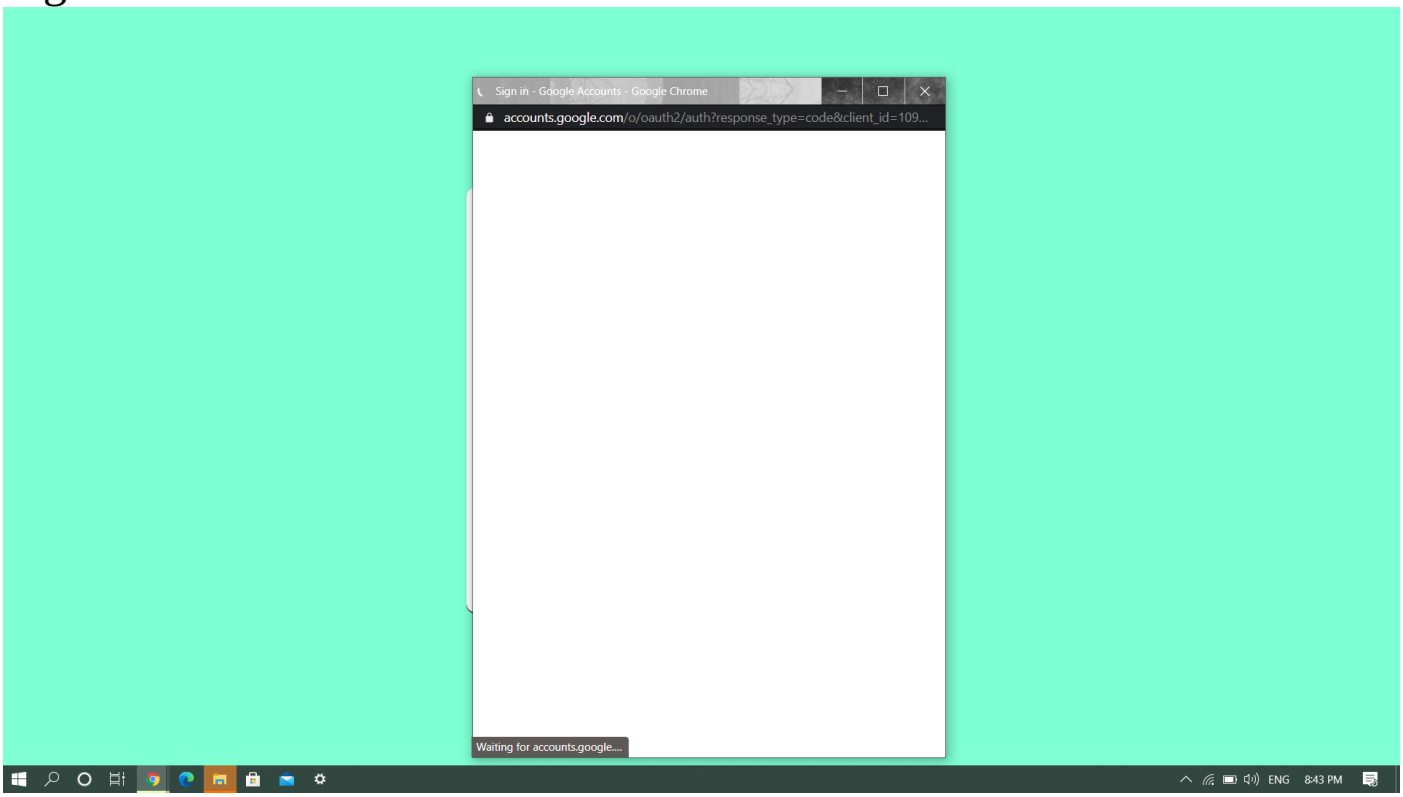
Structure tests are concerned with exercising the internal logic of a program and traversing particular execution paths. Some authors refer collectively to functional performance and stress testing as “black box” testing. While structure testing is referred to as “white box” or “glass box” testing. The major activities in structural testing are deciding which path to exercise, deriving test date to exercise those paths, determining the test coverage criterion to be used, executing the test, and measuring the test coverage achieved when the test cases are exercised.

Chapter-5
Implementation and User Interface

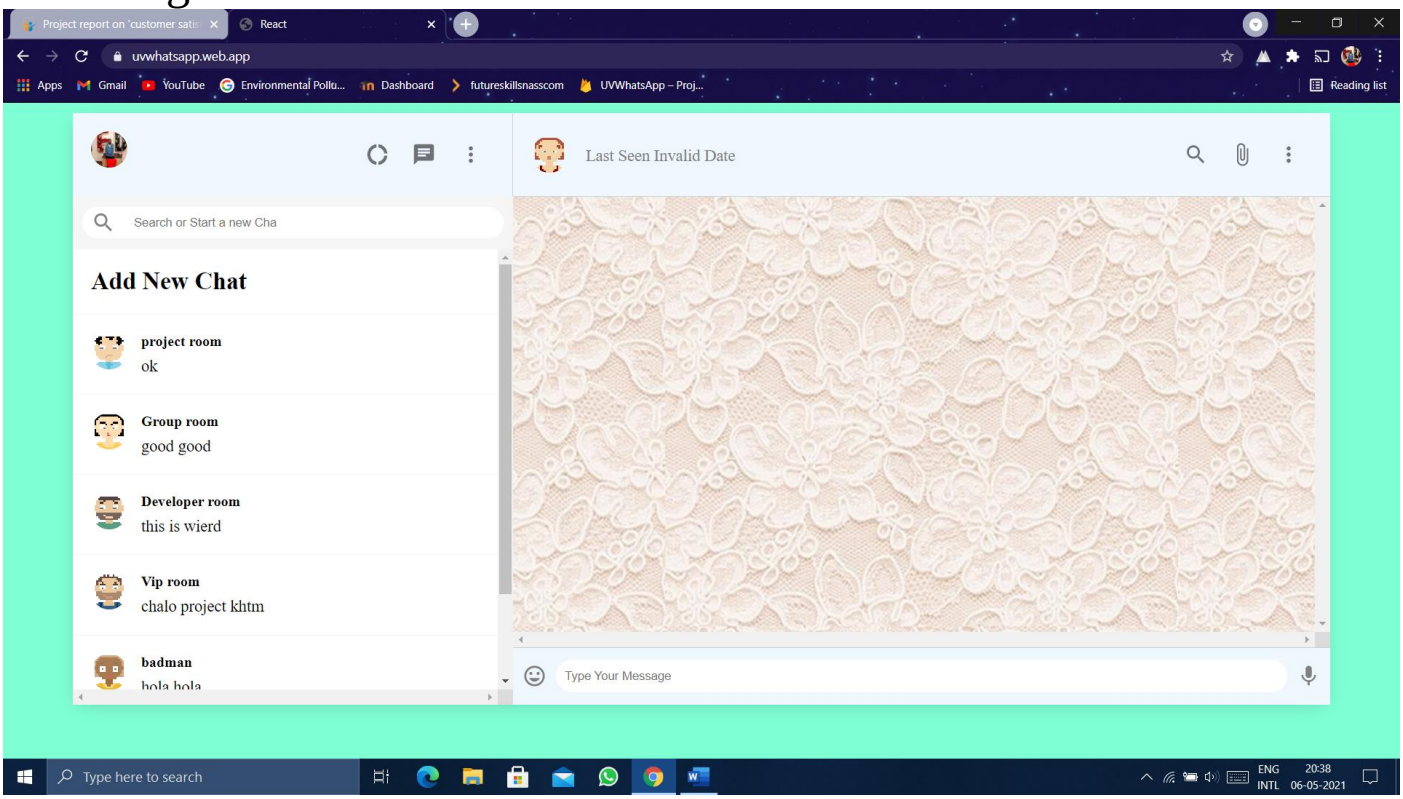
Welcome Page:



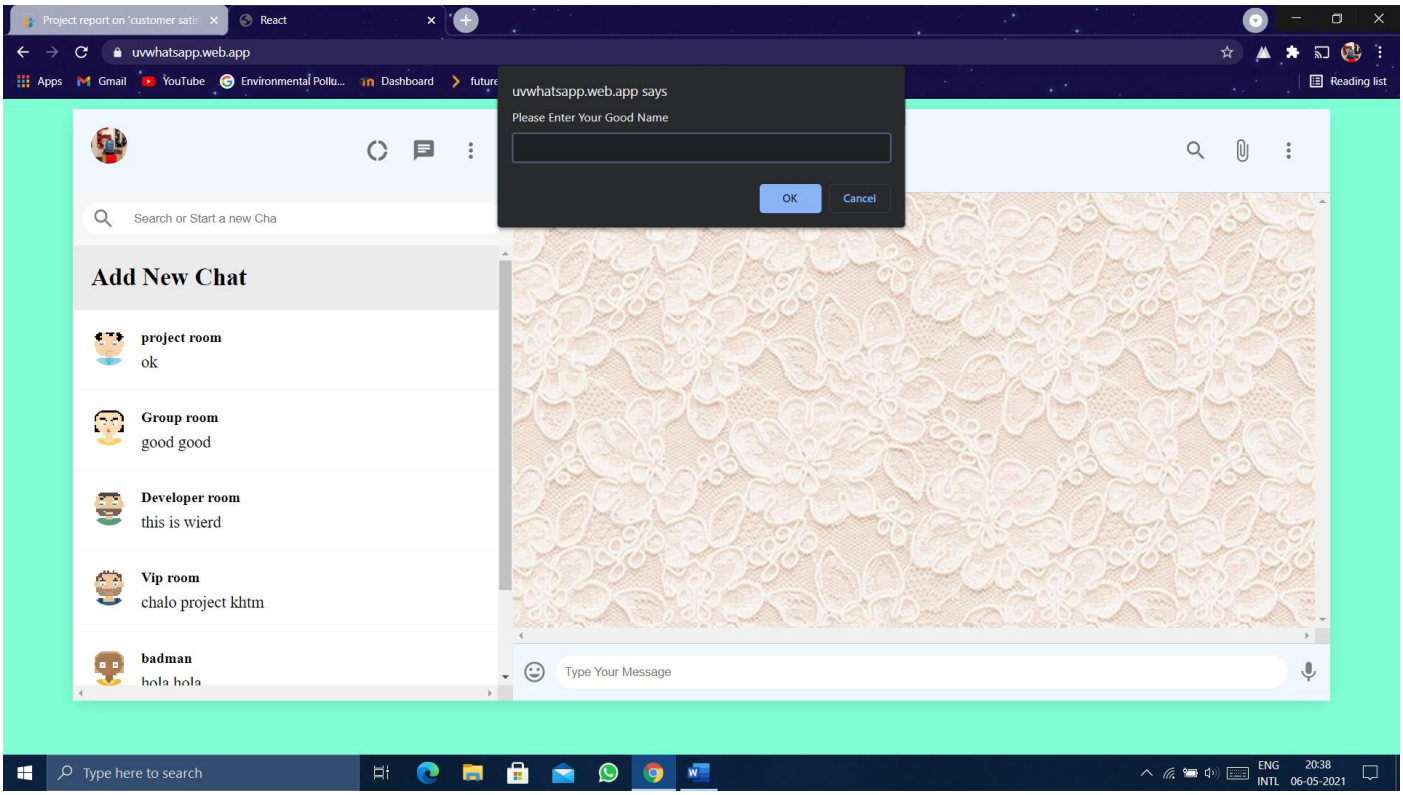
Sign-in Process:



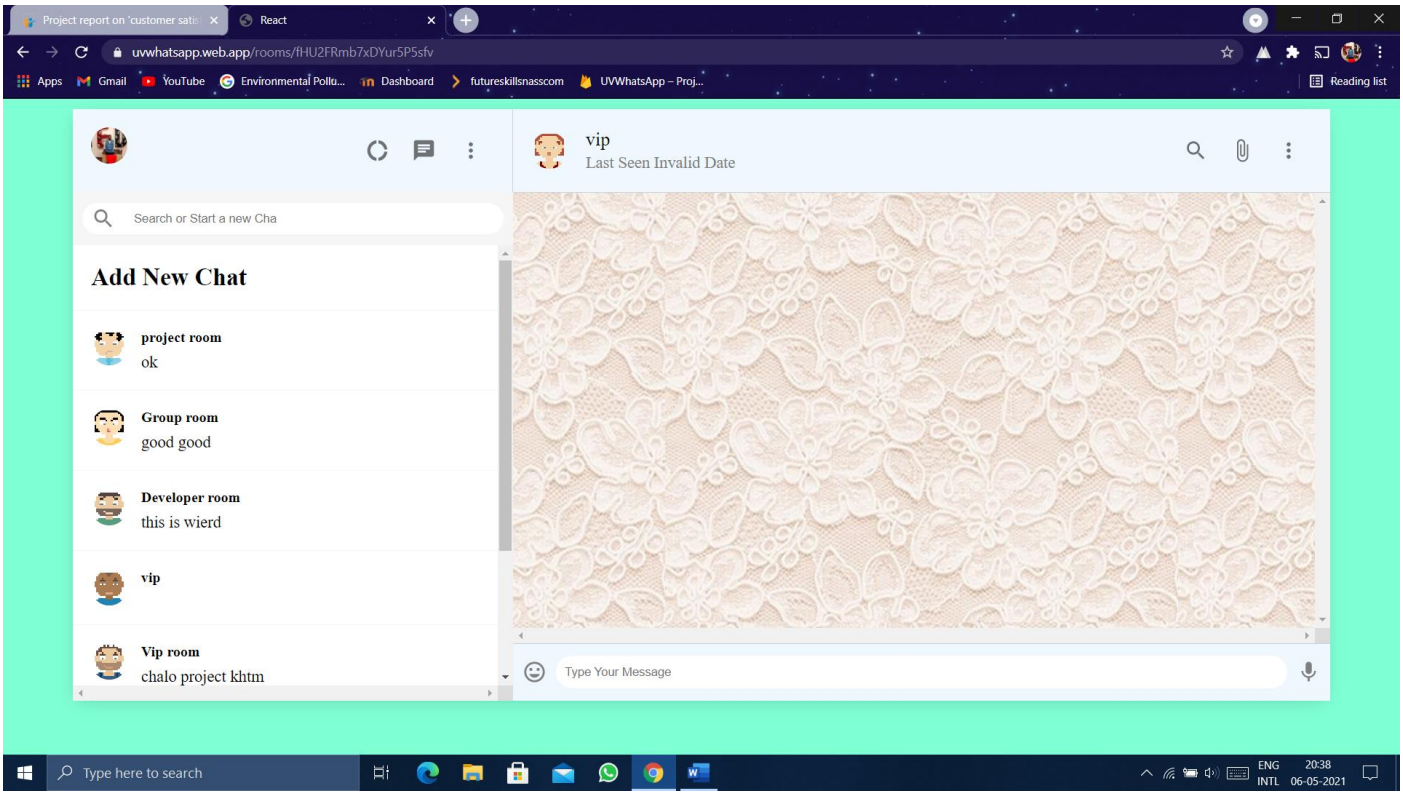
UI of Page:

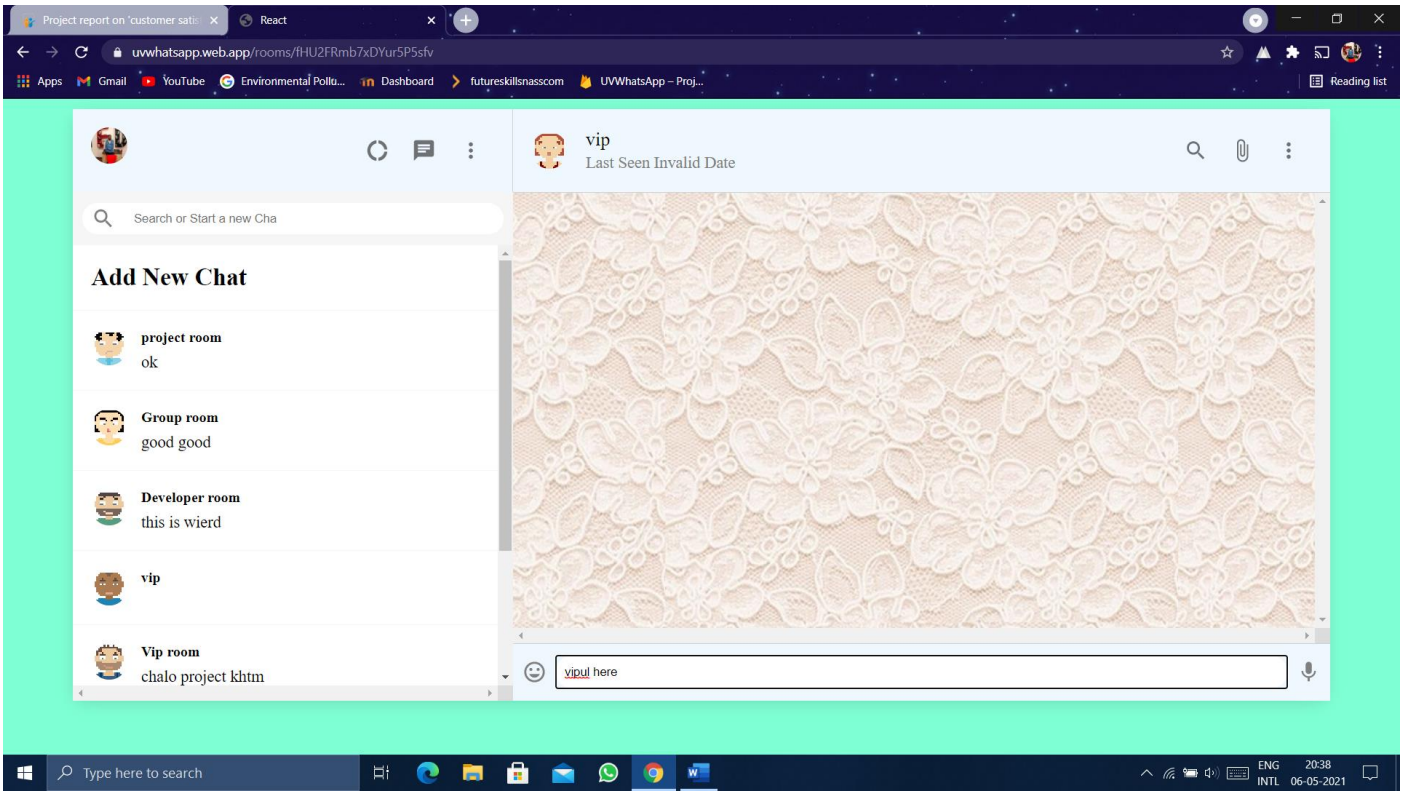


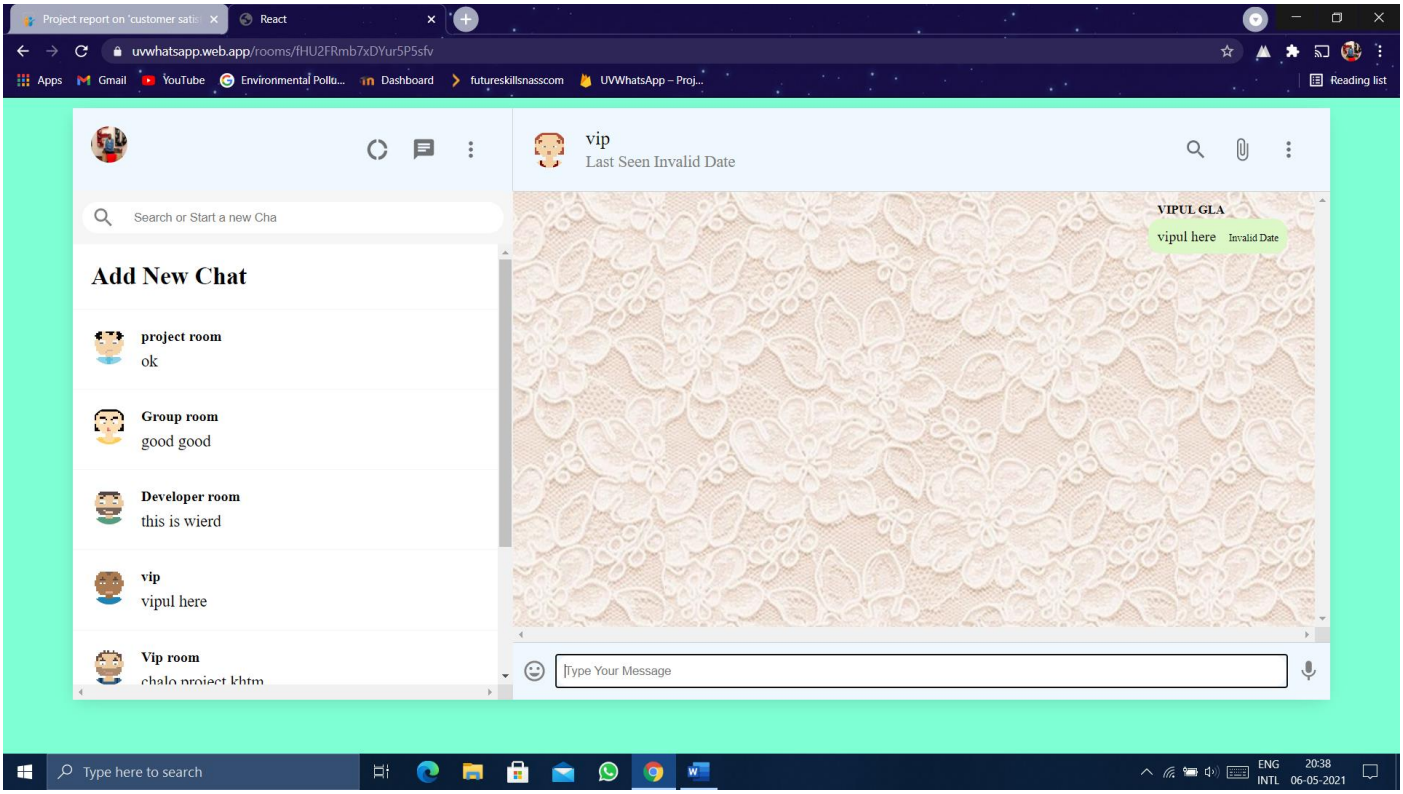
Add New Rooms:

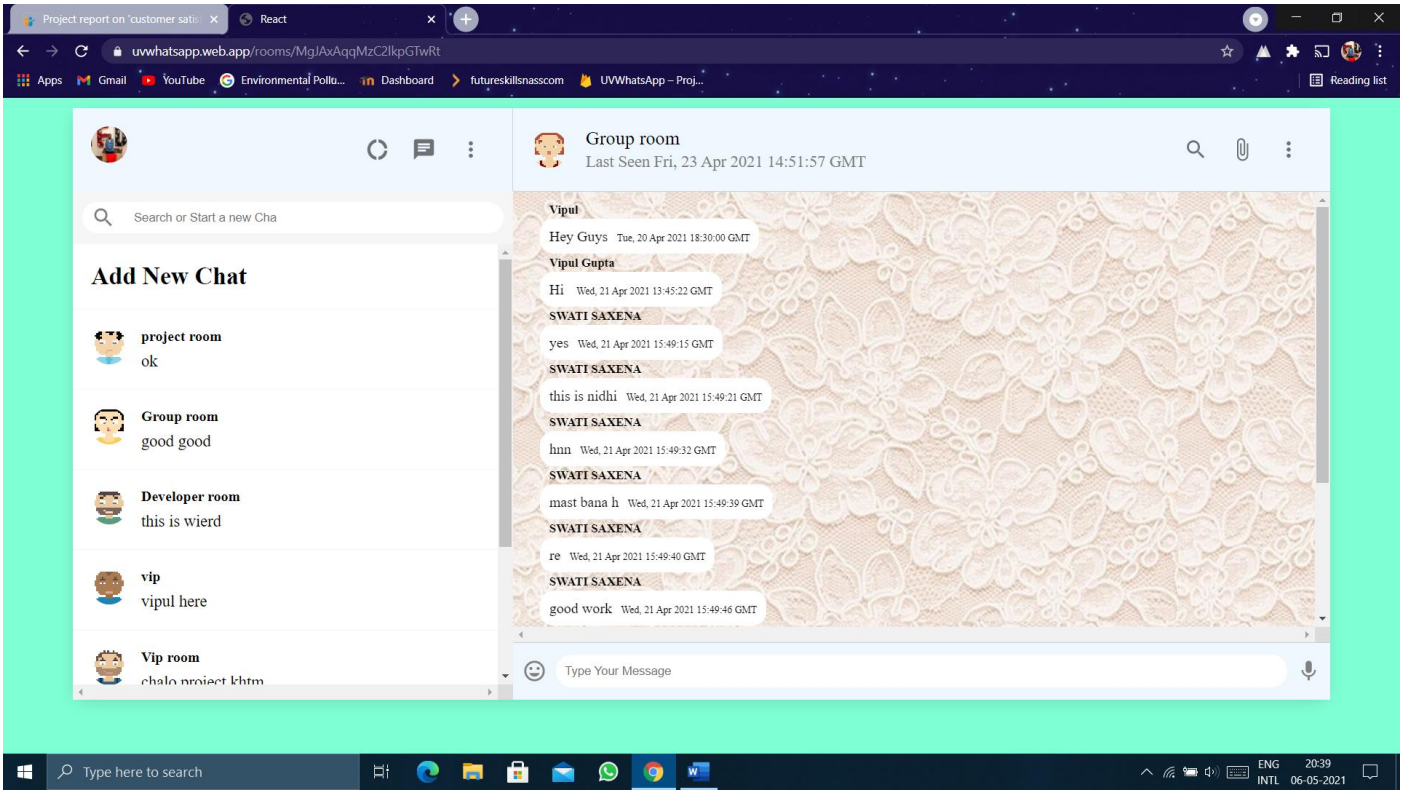


Room Added:


uvwhatsapp.web.app/rooms/fHU2FRmb7xDYur5P5sfv
vip
Last Seen Invalid Date
Search or Start a new Cha
Add New Chat
project room
ok
Group room
good good
Developer room
this is wierd
vip
Vip room
chalo project khtm
Type Your Message
Type here to search


uvwhatsapp.web.app/rooms/fHU2FRmb7xDYur5P5sfv
vip
Last Seen Invalid Date
Search or Start a new Cha
Add New Chat
project room
ok
Group room
good good
Developer room
this is wierd
vip
Vip room
chalo project khtm
vipul here
Type here to search


uvwhatsapp.web.app/rooms/fHU2FRmb7xDYur5P5sfv
vip
Last Seen Invalid Date
Search or Start a new Cha
Add New Chat
project room
ok
Group room
good good
Developer room
this is wierd
vip
vipul here
Vip room
VIPUL GLA
vipul here Invalid Date
Type Your Message
Type here to search


uvwhatsapp.web.app/rooms/MgJAxAqqMzC2lkpGTwRt
Group room
Last Seen Fri, 23 Apr 2021 14:51:57 GMT
Search or Start a new Cha
Add New Chat
project room
ok
Group room
good good
Developer room
this is wierd
vip
vipul here
Vip room
Vipul
Hey Guys Tue, 20 Apr 2021 18:30:00 GMT
Vipul Gupta
Hi Wed, 21 Apr 2021 13:45:22 GMT
SWATI SAXENA
yes Wed, 21 Apr 2021 15:49:15 GMT
SWATI SAXENA
this is nidhi Wed, 21 Apr 2021 15:49:21 GMT
SWATI SAXENA
hnn Wed, 21 Apr 2021 15:49:32 GMT
SWATI SAXENA
mast bana h Wed, 21 Apr 2021 15:49:39 GMT
SWATI SAXENA
re Wed, 21 Apr 2021 15:49:40 GMT
SWATI SAXENA
good work Wed, 21 Apr 2021 15:49:46 GMT
Type Your Message
Type here to search

Chapter-6
Contributions

Harshita Bhardwaj
She contributed the design and frontend designs.

Vipul
He contributed the working of our project.

Umesh Pratap Singh
He contributed the User-Interface and authentication.

Chapter-7
References

- www.javatpoint.com
- www.w3school.com
- www.beta-labs.in
- www.youtube.com
- https://www.youtube.com/watch?v=pUxrDcITyjg&t=12503s

Chapter-8
Appendices

App.js:

```
import React from 'react';
import Sidebar from './Sidebar';
import './App.css';
import Chat from './Chat';
import { Switch, Route } from 'react-router';
import { BrowserRouter as Router} from 'react-router-dom';
import Login from './Login';
import { useStateValue } from './StateProvider';
function App(){
  const [{user}] = useStateValue();
  return(
    <div className="app">
      {!user ? (
        <Login />
      ) : (
        <div className="appbody">
          <Router>
            <Sidebar />
            <Switch>
              <Route path='/rooms/:roomId'>
                <Chat />
              </Route>
              <Route path='/'>
                <Chat />
              </Route>
            </Switch>
          </Router>
        </div>
      )}
    </div>
  );
}
export default App;
```

App.css:

```
.app{
    background-color: aquamarine;
    display: grid;
    place-items:center;
    height: 100vh;
}
.appbody{
```

```
    width: 90vw;
    height: 90vh;
    display: flex;
    margin-top: -50px;
    background-color: aliceblue;
    box-shadow: -1px 4px 20px -6px rgba(0, 0,0,0.2);
}
```

Chat.js:

```
import { Avatar,IconButton } from '@material-ui/core';
import { AttachFile, InsertEmoticon, Mic, MoreVert, SearchOutlined } from '@material-
ui/icons';
import userEvent from '@testing-library/user-event';
import React,{useEffect,useState} from 'react';
import { useParams } from 'react-router';
import './Chat.css';
import db from './Firebase';
import firebase from 'firebase';
import { useStateValue } from './StateProvider';
export default function Chat() {
    const [input,setInput] = useState('');
    const [seed,setSeed] = useState('');
    const {roomId} = useParams();
    const [roomName,setRoomname] = useState('');
    const [messages,setMessages] = useState([]);
    const [{user},dispatch] = useStateValue();

    useEffect(()=>{
        if(roomId){
            db.collection('rooms').doc(roomId).onSnapshot(snapshot => (
                setRoomname(snapshot.data().name)
            ))
            db.collection('rooms').doc(roomId).collection('messages').orderBy('timestamp',
'asc').onSnapshot(snapshot =>(
                setMessages(snapshot.docs.map(doc => doc.data()))
            ))
        }
    },[roomId]);

    useEffect(() => {
        setSeed(Math.floor(Math.random()*5000));
    }, [])
    const sendMessage =(e)=>{
        e.preventDefault();
        db.collection('rooms').doc(roomId).collection('messages').add({
            message:input,
            name:user.displayName,
            timestamp:firebase.firestore.FieldValue.serverTimestamp(),
```

```
        });
        setInput("");
    };
    return (
        <div className="chat">
            <div className="chat_header">
                <Avatar src={`https://avatars.dicebear.com/api/human/${seed}.svg`}/>
                <div className="chat_headerInfo">
                    <h3>{roomName}</h3>
                    <p>
                        Last Seen{" "}
                        {new Date(messages[messages.length-
1]?.timestamp?.toDate()).toUTCString()}
                    </p>
                </div>
                <div className="chat_headerRight">
                    <IconButton>
                        <SearchOutlined/>
                    </IconButton>
                    <IconButton>
                        <AttachFile/>
                    </IconButton>
                    <IconButton>
                        <MoreVert/>
                    </IconButton>
                </div>
            </div>
            <div className="chat_body">
                {messages.map(message =>(
                    <p className={`chat_message ${message.name===user.displayName && 'chat
_receiver'}`}>
                    <span className="chat_name">{message.name}</span>
                    {message.message}
                    <span className="chat_timestamp">{new Date(message.timestamp?.toDate()
).toUTCString()}</span>
                    </p>
                ))}
            </div>
            <div className="chat_footer">
                <InsertEmoticon/>
                <form action="">
                    <input value={input} onChange={(e)=>setInput(e.target.value)} placehol
der="Type Your Message" type="text"/>
                    <button onClick={sendMessage} type="sumbit">Send a Message</button>
                </form>
                <Mic/>
            </div>
        </div>
    )
}
```

Login.js:

```
import React from 'react'
import './Login.css';
import {Button} from '@material-ui/core';
import { auth, provider } from './Firebase';
import { useStateValue } from './StateProvider';
import { actionTypes } from './reducer';
function Login() {
    const [{},dispatch] = useStateValue();
    const signIn = () =>{
        auth.signInWithPopup(provider).then((result) => {
            dispatch({
                type:actionTypes.SET_USER,
                user:result.user,
            })
        }).catch((error)=>alert(error.message));
    };
    return (
        <div className="login">
            <div className="login_container">
                <img src="https://img.icons8.com/cotton/2x/whatsapp--v4.png" alt=""/>
                <div className="login_text">
                    <h1>Signin to UVWhatsapp</h1>
                </div>
                <Button onClick={signIn}>
                    Signin with Google
                </Button>
            </div>
        </div>
    )
}

export default Login
```

Reducer.js:

```
export const initialState={
    user:null,
};
export const actionTypes={
    SET_USER : "SET_USER",
};
const reducer = (state,action)=>{
    console.log(action);
    switch(action.type){
        case actionTypes.SET_USER:
            return{
                ...state,
```

```
                user:action.user,
            };
            default:
                return state;
    }
};

export default reducer;
```

Sidebar.js:

```
import { Avatar, IconButton } from '@material-ui/core';
import { Chat, DonutLarge, MoreVert, SearchOutlined } from '@material-ui/icons';
import React, { useEffect, useState } from 'react';
import './Sidebar.css';
import SidebarChat from './SidebarChat';
import db from './Firebase';
import { useStateValue } from './StateProvider';

function Sidebar(){
  const [rooms,setRooms] = useState([]);
  const [{user},dispatch] = useStateValue();
  useEffect(()=>{
    const unsubscribe = db.collection('rooms').onSnapshot(snapshot =>(
      setRooms(snapshot.docs.map(doc =>({
        id:doc.id,
        data:doc.data(),
      })))
    ));
    return () =>{
      unsubscribe();
    }
  },[]);
  return(
    <div className="sidebar">
      <div className="sidebar_header">
        <Avatar src={user?.photoURL}/>
        <div className="sidebar_headerRight">
            <IconButton>
                <DonutLarge/>
            </IconButton>
            <IconButton>
                <Chat/>
            </IconButton>
            <IconButton>
                <MoreVert/>
            </IconButton>
        </div>
      </div>
      <div className="sidebar_search">
```

```
        <div className="sidebar_searchContainer">
            <SearchOutlined/>
            <input placeholder="Search or Start a new Chat" type="text"/>
        </div>
      </div>
      <div className="sidebar_chats">
        <SidebarChat addnewchat />
        {rooms.map(room =>(
          <SidebarChat key={room.id} id={room.id} name={room.data.name} />
        ))}
      </div>
    </div>
  );
}

export default Sidebar;
```

index.js:

```
import React from 'react';
import ReactDOM from 'react-dom';
import App from './App';
import './index.css';
import reducer, {initialState} from './reducer';
import {StateProvider} from './StateProvider';
ReactDOM.render(
  <React.StrictMode>
    <StateProvider initialState={initialState} reducer={reducer}>
      <App />
    </StateProvider>
  </React.StrictMode>,
  document.getElementById('root')
);
```

Chat.css:

```
.chat{
    flex: 0.65;
    display: flex;
    flex-direction: column;
}
.chat_header{
    display: flex;
    padding: 20px;
    align-items: center;
    border-bottom: 1px solid lightgray;
}
.chat_message{
    position: relative;
    padding: 10px;
```

```
    font-size: 16px;
    background-color: #ffffff;
    width: fit-content;
    border-radius: 15px;
    margin-bottom: 20px;
}
.chat_name{
    position: absolute;
    top: -18px;
    font-weight: 800;
    font-size: x-small;
}
.chat_timestamp{
    font-size: xx-small;
    margin-left: 10px;
}
.chat_receiver{
    margin-left: auto;
    background-color: #dcf8c6;
}
.chat_headerInfo{
    flex: 1;
    padding-left: 20px;
}
.chat_headerInfo>h3{
    font-weight: 500;
    margin-bottom: 3px;
}
.chat_headerInfo>p{
    color: gray;
}
.chat_headerRight{
    display: flex;
    justify-content: space-between;
    min-width: 100px;
}
.chat_body{
    background-image: url('https://1.bp.blogspot.com/-
RxFZX0PBISw/VwChX8KAhAI/AAAAAAAAXIw/e462l-
cYFrAmmLQBZvwp17p01Ml8tM6yw/s1600/Whatsapp%2BWallpapers%2Band%2BBackgrounds%2B%252816%2529
.jpg');
    flex: 1;
    background-repeat: repeat;
    background-position: center;
    padding: 30px;
    overflow: scroll;
}
.chat_footer{
    height: 62px;
    border-top: 1px solid lightgray;
```

```
    display: flex;
    align-items: center;
    justify-content: center;
}
.chat_footer>form{
    flex: 1;
    display: flex;
}
.chat_footer>form>input{
    flex: 1;
    border-radius: 30px;
    padding: 10px;
    border: none;
}
.chat_footer>form>button{
    display: none;
}
.chat_footer>.MuiSvgIcon-root{
    padding: 10px;
    color: gray;
}
```

Login.css:

```
.login{
    height: 100vh;
    width: 100vh;
    display: grid;
    place-items: center;
    background-color: aquamarine;
}
.login_container{
    padding: 100px;
    text-align: center;
    border-radius: 10px;
    background-color: white;
    box-shadow: 0 1px 3px rgba(0,0,0,0.12), 0 1px;
}
.login_container>img{
    object-fit: contain;
    height: 100px;
    margin-bottom: 40px;
}
.login_container>Button{
    text-transform: inherit !important;
    background-color: #0a8d0a !important;
    color: white;
    margin-top: 50px;
}
```

Sidebar.css:

```
.sidebar{
    display: flex;
    flex-direction: column;
    flex: 0.35;
}
.sidebar_header{
    display: flex;
    justify-content: space-between;
    padding: 20px;
    border-right: 1px solid lightgrey;
}
.sidebar_headerRight{
    display: flex;
    justify-content: space-between;
    align-items: center;
    min-width: 10vw;
}
.sidebar_headerRight> .MuiSvgIcon-root{
    margin-right: 2vw;
    font-size: 24px !important;
}
.sidebar_search{
    display: flex;
    align-items: center;
    background-color: #f6f6f6;
    height: 39px;
    padding: 10px;
}
.sidebar_searchContainer{
    display: flex;
    align-items: center;
    background-color: white;
    height: 35px;
    width: 100%;
    border-radius: 20px;
}
.sidebar_searchContainer>input{
    border: none;
    margin-left: 10px;
}
.sidebar_searchContainer>.MuiSvgIcon-root{
    color: grey;
    padding: 10px;
}
.sidebar_chats{
    flex: 1;
    background-color: white;
```

```
    overflow: scroll;
}
```

StateProvider.js:

```
import React, { createContext,useReducer,useContext } from 'react';

export const StateContext = createContext();
export const StateProvider = ({reducer,initialState,children}) => (
    <StateContext.Provider value= {useReducer(reducer,initialState)}>
        {children}
    </StateContext.Provider>
);
export const useStateValue = () => useContext(StateContext);
```