A

Project Report On

"Offline Chat Application"

By

More Pratik (38)
Nehul Krushna (41)
Parth Pidurkar (46)
Pokharna Jay (47)
Selot Gautam (53)

Guided By
Ms. Nisha Thokal

Department Of
Computer Technology
Government Polytechnic, Ahmednagar

2020-2021

CERTIFICATE

This is to certify that the project entitled "**Offline Chat Application**" has been carried out been

More Pratik	(38)
Nehul Krushna	(41)
Parth Pidurkar	(46)
Pokharna Jay	(47)
Selot Gautam	(53)

Under my guidance in partial fulfillment of the diploma of Computer Technology of Government Polytechnic, Ahmednagar during academic year 2020-2021 to the best of knowledge and belief. This work has not seen submitted elsewhere for the award of any other diploma.

GUIDE	H. O. D.	PRINCIPAL
(Ms Nisha Thokal)	(Mr.S.D.Muley)	(Mr.M.S.Satarkar)

INDEX

Acknowledgement

We take this opportunity to thank our guide Ms. Nisha Thokal for placing this idea in our mind and giving marvelous suggestions from the platform of this project. We are pleased to take the opportunity to construct a project **"Offline Chat Application"**. The project title would not have been completed without the valuable guidance and encouragement of my family and friends.

Offline Chat Application

1.0 Rationale:-

Nowadays all people are interested in chat systems. They want to connect with people all over the world from their home with privacy and security. Therefore a secured system is very essential. This project is developed for everyone, so any one can used this application. We can handle this software easily.

2.0 Aims/Benefits of the Micro-Project:-

The Aim of this project is to develop software for "Offline Chat System". We can connect peoples without any internet connection. Children as well as adult people who has basic knowledge about chat systems can use it. It will be fun to chat with people using graphical user interface.

3.0 Course Outcomes Integrated:-

- Create interactive web pages using program flow control structure.
- Implement Array and functions in Java script.
- Create event based web forms using Java script.

4.0 Literature Review:-

We have created interacting GUI using HTML and CSS. For our project, firstly we have to know about basic of HTML to display the content on browser. Then we have required basic of CSS to add some style to our content like color, size, background etc. We use Javascript to do validation. The 'Socket.io' library is used to enables realtime, bi-directional communication between web clients and server. We have to know about the working of different functions of Socket.io library like createServer()-To create server, socket.broadcast.emit()-To send message to the client etc.

5.0 Actual Methodology Followed:-

1. Firstly, all team members together decided an appropriate topic.
2. Then Pratik More and Selot Gautam did the analysis of available resources.
3. Following it, searching of information on the application was done by Nehul Krushna and Pidurkar Parth.
4. Then all group member start the codding of application.
5. Now firstly we designed the GUI sample in paint.
6. Our Chat Application GUI includes App Logo, user's name, setting button, chat area, typing area, voice call button, video call button, dark and light theme.

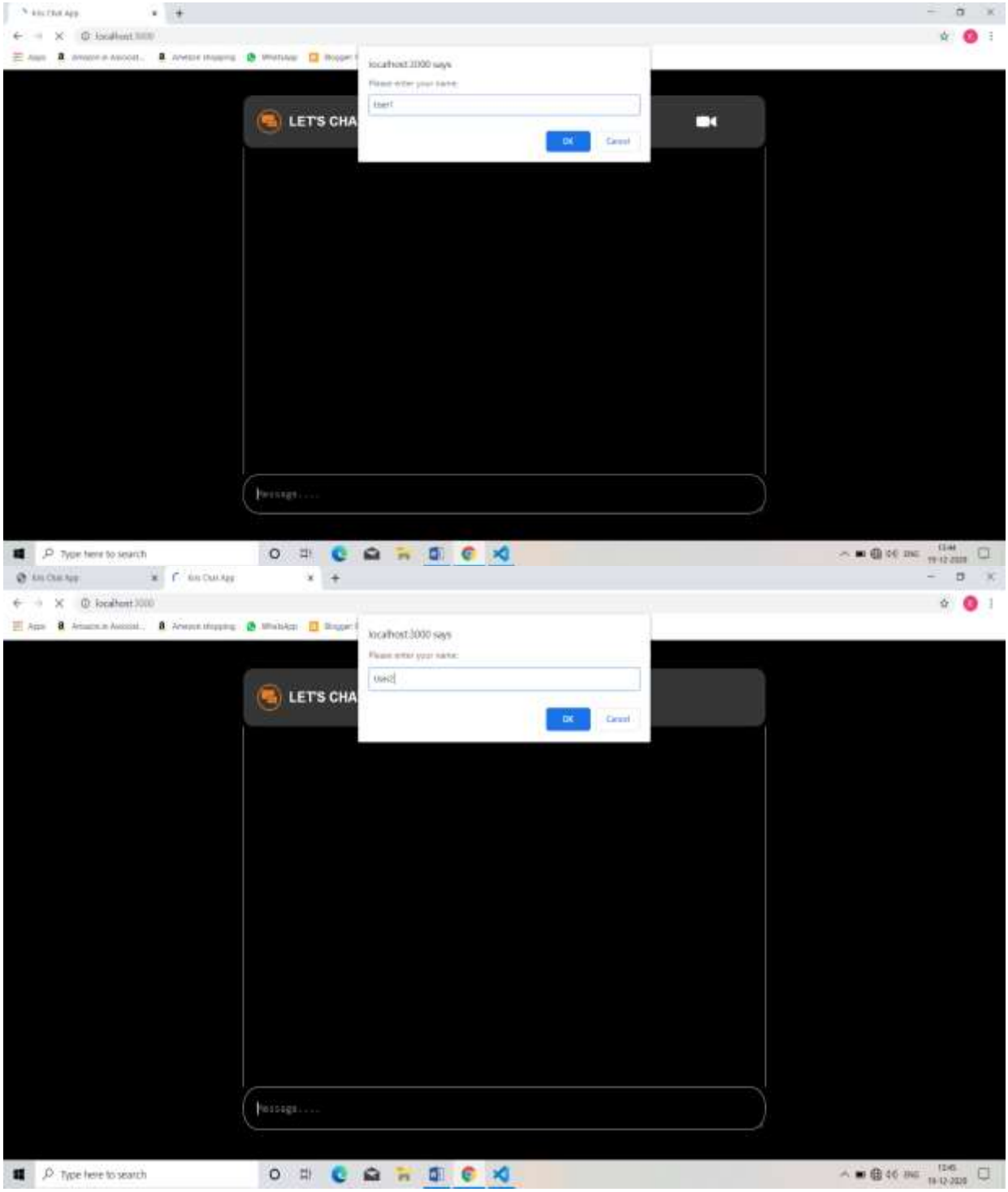
7. So we started creating GUI for client side using html and css. Then we added the light and dark theme options.
8. Then we add voice call button and video call button in the window.
9. Then we created Server using ‘Socket.io’ libraries methods like createServer().
10. Then we write script for client side to send message to the server.
11. In client side we create a function to append the message in the message area.
12. Now here the main part of the application is ready i.e. the messages can be shared between server and client.
13. Now we added the theme setting button to change the theme between Dark and Light.
14. At last we add voice and video call buttons.
15. Here our programming ends and in this way, we successfully built an Offline Chat Application.

6.0 Actual Resources Used:-

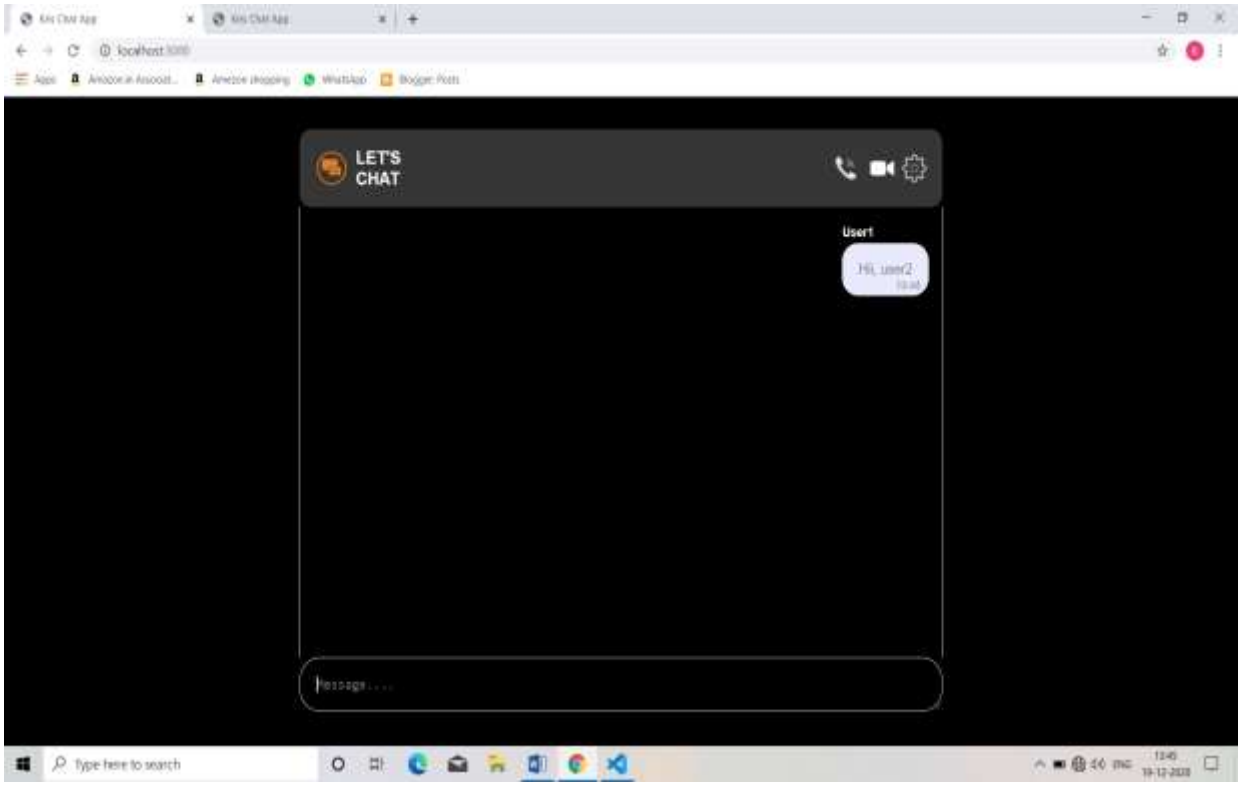
Sr. No.	Resources Required	Specifications	Quantity	Remark
1	Hardware Desktop PC	Intel Dual Core i3 Ram – 12GB Storage – 300GB	1	
2	Operating System	Windows 10	1	
3	Software	Visual Studio Code, node.js, Ms-Word 13	1	

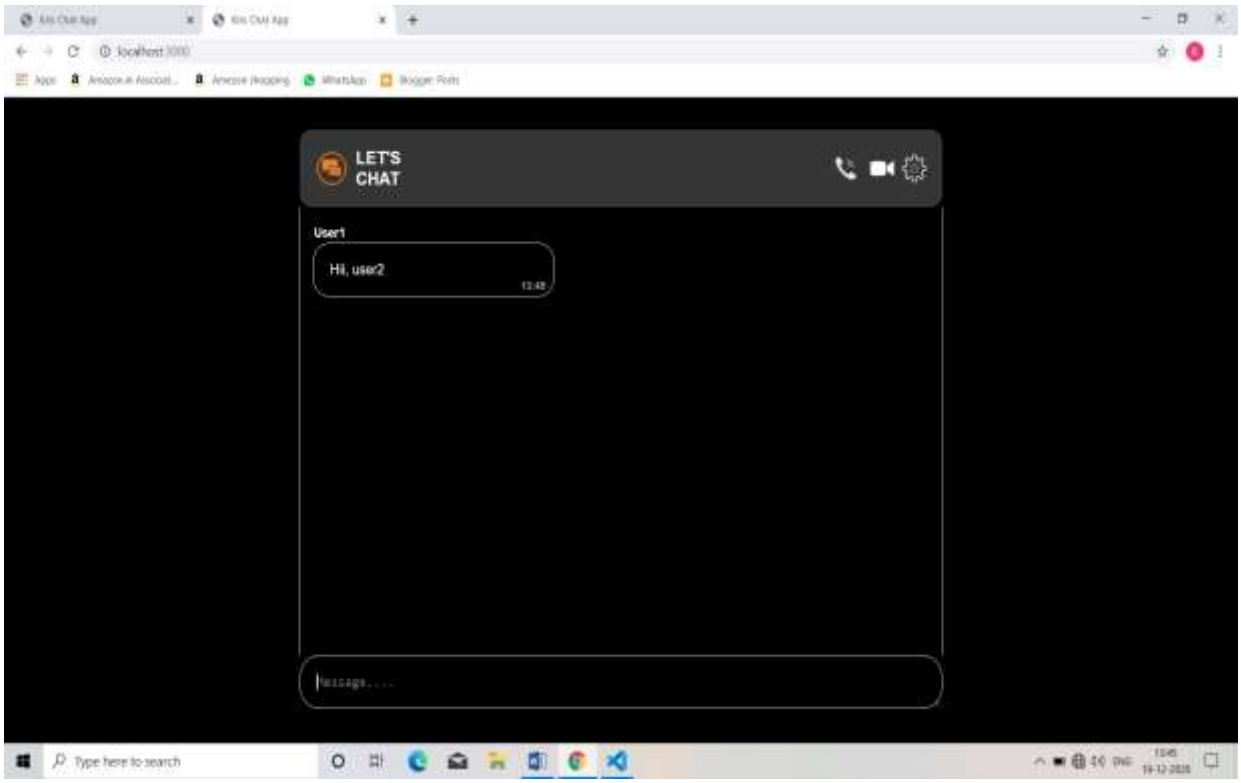
7.0 Outputs of the Micro-Project:-

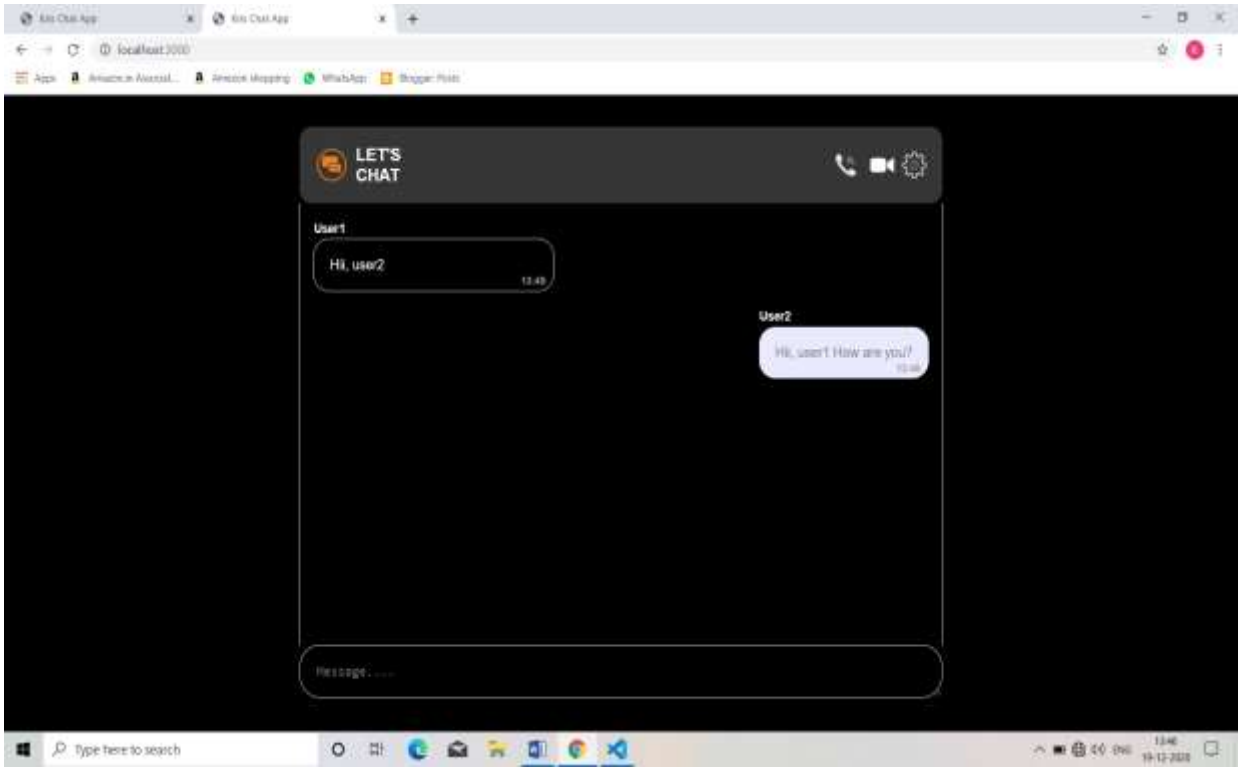
Entering Name of user-

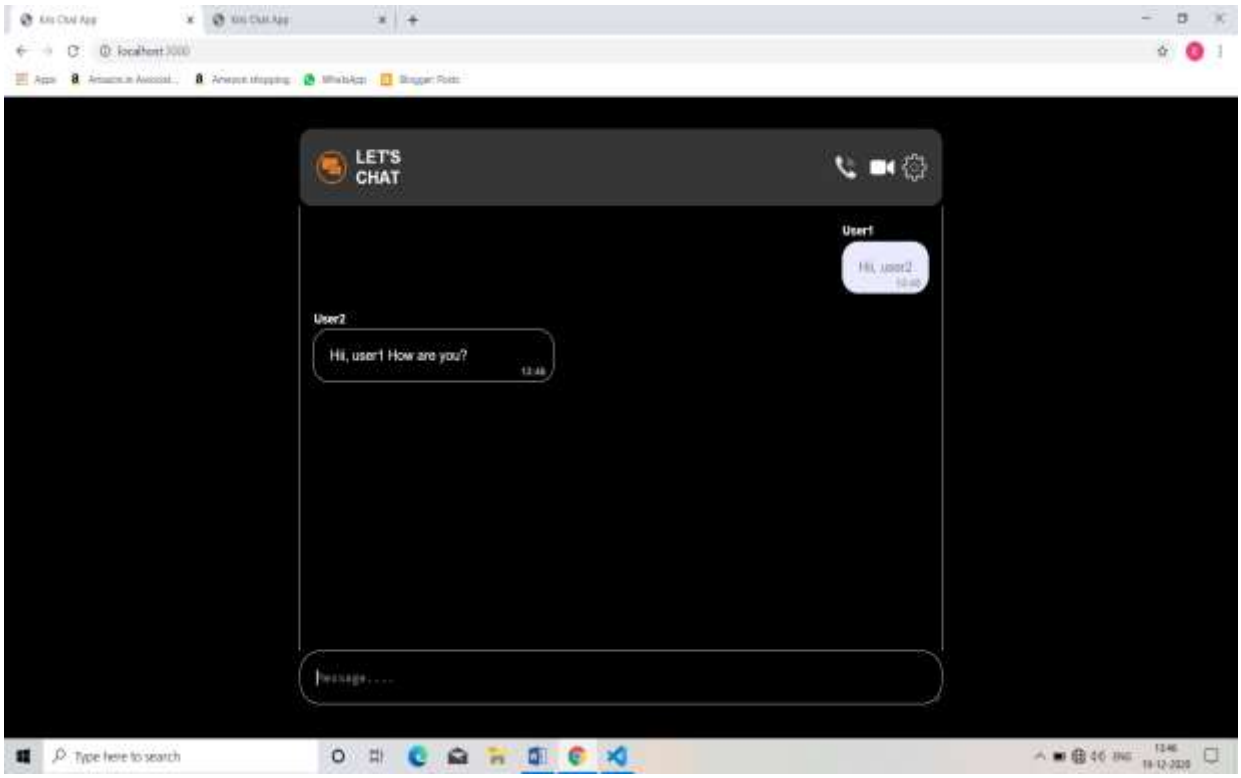


Sending message to other users-

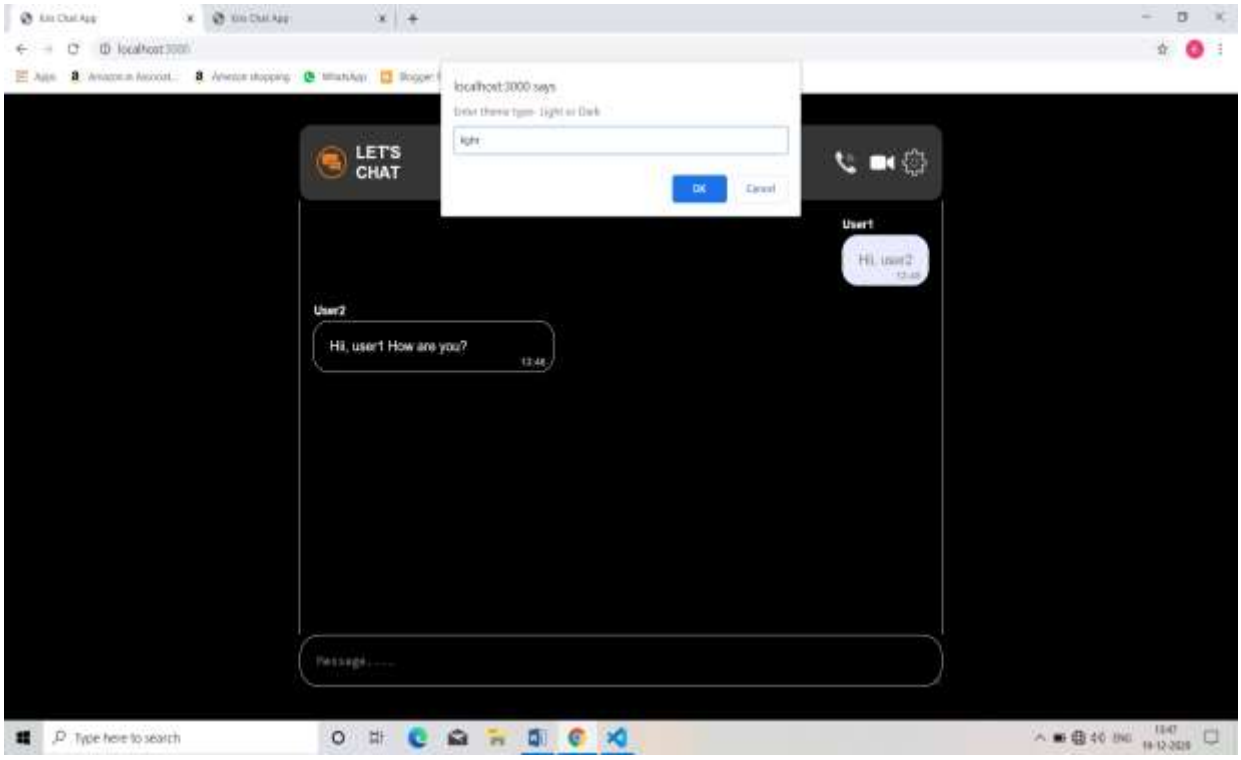


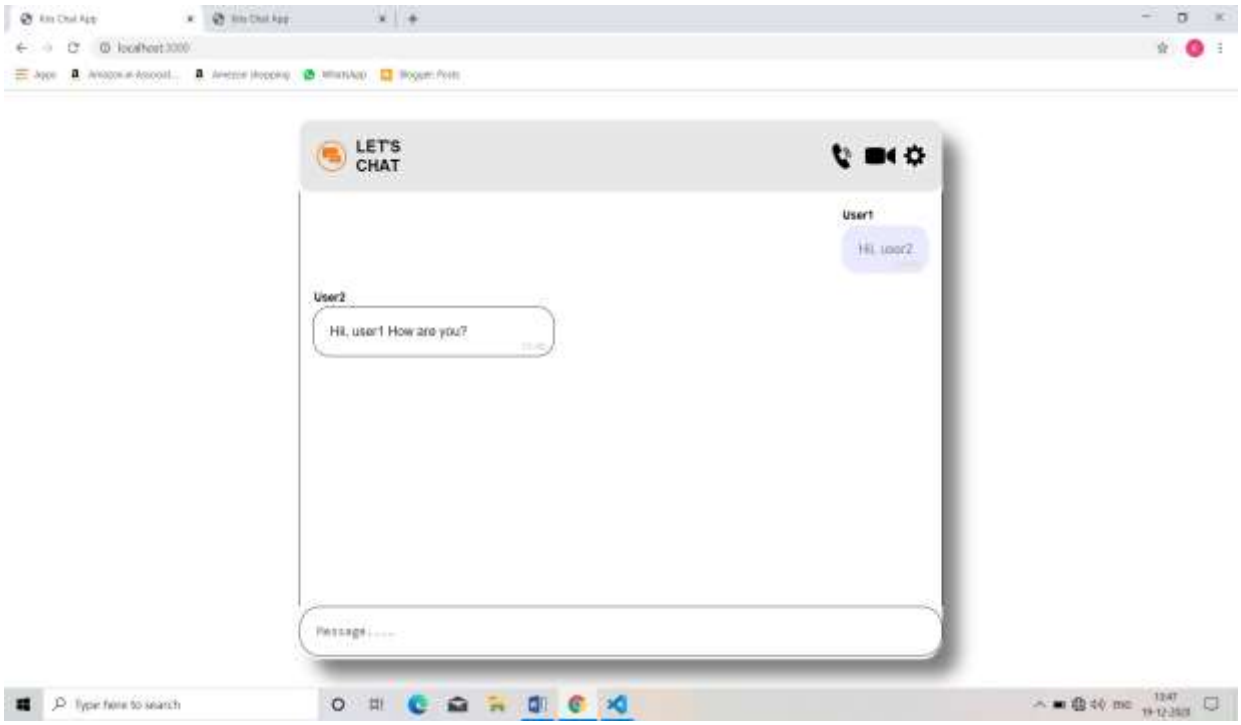

LET'S
CHAT
User1
Hi, user2
Message....
Type here to search


LET'S
CHAT
User1
Hi, user2
User2
Hi, user1 How are you?
Message....
Type here to search

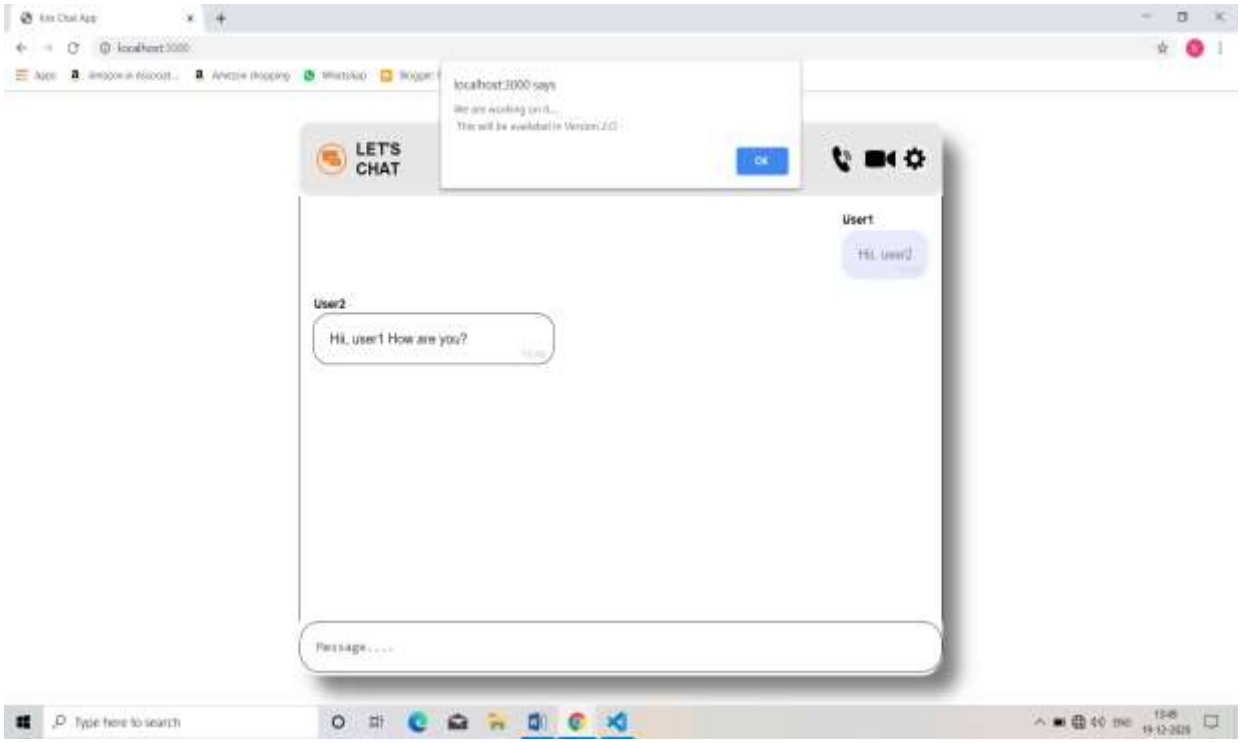


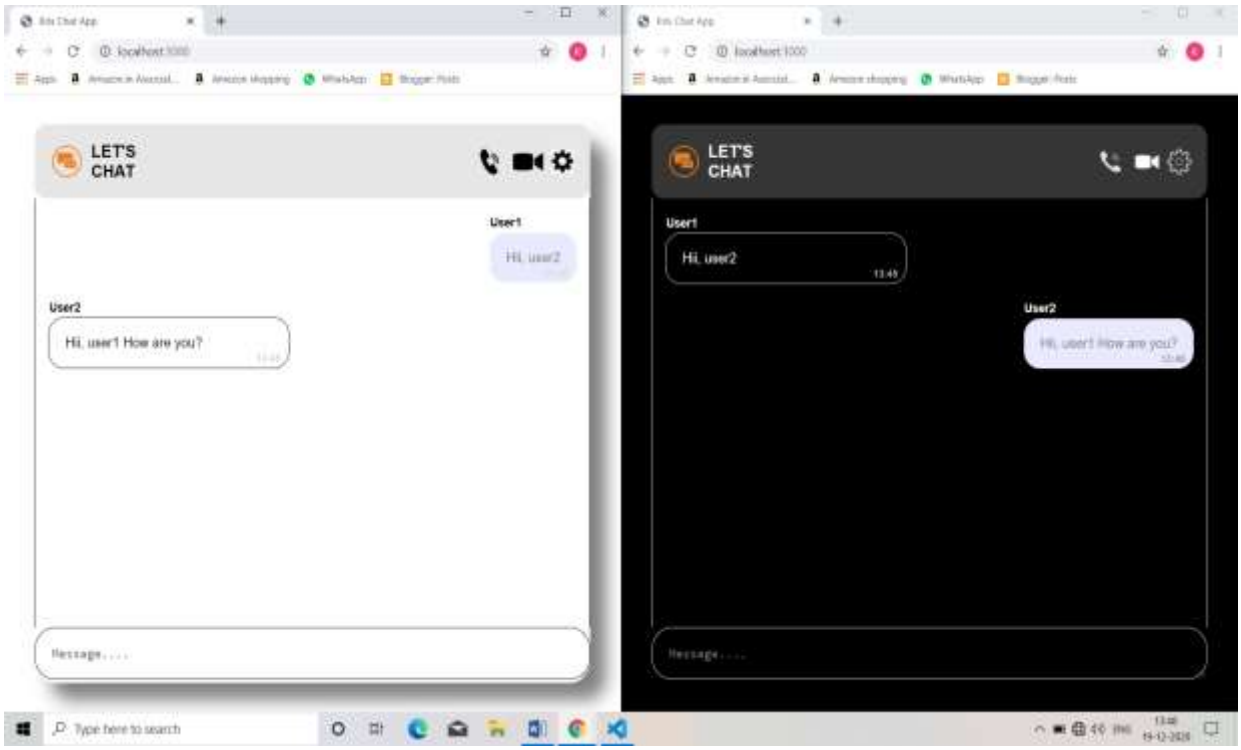
Changing the theme-





Trying to call-




LET'S CHAT
User1
Hi, user2
User2
Hi, user1 How are you?
Message....
LET'S CHAT
User1
Hi, user2
User2
Hi, user1 How are you?
Message....

8.0 Skills Developed / Learning out of this Micro Project:-

We learned to-

a) Create interactive web pages using program flow control structure.

b) Implement functions in Java script.

c) Create event based web forms using Java script.

9.0 Application of this micro project:-

- Our Offline chat application provides communication service which enables the user to send messages in the absence of internet connection.
- It can be used in the areas with bad/poor network connectivity.
- This chat application is very use full in offices to share media between multiple users without the requirement of internet connection.
- All the messages and other media shared through this technology is completely secured and no issues like hacking is possible as you are directly connected to the other user.

10.0 Area of Future improvement:-

- We can develop our application more by adding the Bluetooth technology to it which can increase the range of connectivity between the users.
- Video call and voice call features can be built in the application using Bluetooth technology.
- We can even create multiple user environment to provide connectivity within a group of users.

11.0 Program Code:-

I. index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=, initial-scale=1.0">
    <title>Kris Chat App</title>
    <link rel="stylesheet" href="/style.css">
    <script language="javascript">
        function theme()
        {
            theme_type=prompt("Enter theme type- Light or Dark");
            if (theme_type.toLowerCase()==="light") {
                var oldlink = document.getElementsByTagName("link").item('0');
```

```
            oldlink.href = '/style1.css';
        } else if(theme_type.toLowerCase()==="dark") {
            var oldlink = document.getElementsByTagName("link").item('0');
            oldlink.href = '/style.css';
        }
        else{
            alert("Try Again....")
        }
    }
    function inProgress()
    {
        alert("We are working on it....\n This will be availabel in Version 2.O");
    }

  </script>
</head>
<body>
  <section class="chat_section">
    <div class="brand">
      <img height="40"src="/app_logo.png" alt="">
      <h1>Let's Chat</h1>
            <img class="call" onclick="inProgress()">
            <img class="vcall" onclick="inProgress()">
            <img class="setting" onclick="theme()">


    </div>
    <div class="message_area" >

    </div>
    <div>
      <textarea id="textarea" cols="30" rows="1" placeholder="Message...."  autofocus><
/textarea>
    </div>

  </section>
  <script src="/socket.io/socket.io.js"></script>
  <script src="/client.js"></script>
</body>
</html>
```

II. Style1.css

```
@import
    url('https://fonts.googleapis.com/css2?family=Merriweather+Sans&display=swap');
*{
  padding: 0;
  margin: 0;
  box-sizing: border-box;
}
```

```
body{
    display: flex;
    align-items: center;
    justify-content: center;
    min-height: 100vh;
    background: #ffffff;
    font-family: 'Merriweather Sans', sans-serif;
}
section.chat_section{
    width: 800px;
    max-width: 90%;
    background: #ffffff;
    background-size: cover;
    box-shadow:20px 20px 20px rgba(0, 0, 0, 0.5);
    border-radius: 25px ;
}
.brand{
    padding: 20px;
    background: #e7e7e7;
    display: flex;
    align-items: center;
    border-radius: 15px ;
}
.brand h1{
    text-transform: uppercase;
    font-size: 20px;
    color: rgb(0, 0, 0);
    margin-left: 10px;
}
.setting{
    height:30px;
    margin-left: 10px;
    content : url("/setting_L.png");
}
.vcall{
    height:20px;
    margin-left: 15px;
    content : url("/vcall_L.png");
}
.call{
    height:30px;
    margin-left: 65%;
    content : url("/call_L.png");
}
.message_area{
    border-left: 1px solid rgb(0, 0, 0);
    border-right: 1px solid rgb(0, 0, 0);
    height: 500px;
    padding: 16px;
    display: flex;
```

```
    flex-direction: column;
    overflow-y: auto;
    padding-top: 40px;
}
textarea{

    width: 100%;
    border: 1px solid rgb(0, 0, 0);
    padding: 20px;
    font-size: 16px;
    outline: none;
    background: #ffffff;
    border-radius: 25px ;
    color: white;
}
textarea{
    color:black ;
}

.message{
    padding: 20px;
    margin-bottom: 40px;
    max-width: 300px;
    position: relative;
}
.incoming{
    border: 0.5px solid rgb(0, 0, 0);
    border-radius: 20px;
    background: #ffffff;
    color: rgb(0, 0, 0);
    word-wrap: break-word;

}
.outgoing{
    border-radius: 20px;
    background: #e9eafd;
    color: #787986;
    margin-left: auto;
    word-wrap: break-word;

}
.message h4{
    position: absolute;
    top: -20px;
    left: 0;
    color: rgb(0, 0, 0);
    font-size: 14px;
}
.message h6{
```

```
    position: absolute;
    bottom: 5px;
    right: 10px;
    color: rgb(219, 219, 219);
    font-size: 12px;
     }
```

III. Style.css

```
@import
     url('https://fonts.googleapis.com/css2?family=Merriweather+Sans&display=swap');
*{
    padding: 0;
    margin: 0;
    box-sizing: border-box;
}

body{
    display: flex;
    align-items: center;
    justify-content: center;
    min-height: 100vh;
    background: #ffffff;
    font-family: 'Merriweather Sans', sans-serif;
}
section.chat_section{
    width: 800px;
    max-width: 90%;
    background: #ffffff;
    background-size: cover;
    box-shadow:20px 20px 20px rgba(0, 0, 0, 0.5);
    border-radius: 25px ;
}
.brand{
    padding: 20px;
    background: #e7e7e7;
    display: flex;
    align-items: center;
    border-radius: 15px ;
}
.brand h1{
    text-transform: uppercase;
    font-size: 20px;
    color: rgb(0, 0, 0);
    margin-left: 10px;
}
.setting{
    height:30px;
    margin-left: 10px;
    content : url("/setting_L.png");
}
```

```
.vcall{
    height:20px;
    margin-left: 15px;
    content : url("/vcall_L.png");
}
.call{
    height:30px;
    margin-left: 65%;
    content : url("/call_L.png");
}
.message_area{
    border-left: 1px solid rgb(0, 0, 0);
    border-right: 1px solid rgb(0, 0, 0);
    height: 500px;
    padding: 16px;
    display: flex;
    flex-direction: column;
    overflow-y: auto;
    padding-top: 40px;
}
textarea{

    width: 100%;
    border: 1px solid rgb(0, 0, 0);
    padding: 20px;
    font-size: 16px;
    outline: none;
    background: #ffffff;
    border-radius: 25px ;
    color: white;
}
textarea{
    color:black ;
}

.message{
    padding: 20px;
    margin-bottom: 40px;
    max-width: 300px;
    position: relative;
}
.incoming{
    border: 0.5px solid rgb(0, 0, 0);
    border-radius: 20px;
    background: #ffffff;
    color: rgb(0, 0, 0);
    word-wrap: break-word;

}
.outgoing{
```

```
    border-radius: 20px;
    background: #e9eafd;
    color: #787986;
    margin-left: auto;
    word-wrap: break-word;

}
.message h4{
    position: absolute;
    top: -20px;
    left: 0;
    color: rgb(0, 0, 0);
    font-size: 14px;
}
.message h6{

    position: absolute;
    bottom: 5px;
    right: 10px;
    color: rgb(219, 219, 219);
    font-size: 12px;
     }
```

IV. Server.js

```
const express = require('express') //framework for js
const app = express()
const http = require('http').createServer(app)
const PORT = 3000

http.listen(PORT, () => {
    console.log(`Listening on port ${PORT}`)
})
app.use(express.static(__dirname+'/public'))

app.get('/', (req, res) =>{
    res.sendFile(__dirname +'/index.html')
})

//socket

const io = require('socket.io')(http)
io.on('connection',(socket) => {
    console.log('connected...')
    socket.on('message',(msg) =>{
        socket.broadcast.emit('message',msg)
    })
     })
```

V. Client.js

```
const socket = io()
```

```
var name;
let textarea = document.querySelector('#textarea')
let messageArea = document.querySelector('.message_area')
do{
    name=prompt('Please enter your name:')
}while(!name)
textarea.addEventListener('keyup',(e) =>{
    if(e.key ==='Enter'){
        sendMessage(e.target.value)
    }
})

function sendMessage(message){
    let msg = {
        user: name,
        message: message.trim()
    }

    //apend
    appendMessage(msg,'outgoing')
    textarea.value=""
    scrollToBottom()
    //send msg to server
socket.emit('message',msg)
}


function appendMessage(msg,type){
    let mainDiv= document.createElement('div')
    let className = type
    mainDiv.classList.add(className,'message')
    var date=new Date()
    var hours=date.getHours()
    var min=date.getMinutes()
    var time=hours+":"+min
    let markup =`
        <h4>${msg.user}</h4>
        <p>${msg.message}</p>
        <h6>${time}</h6>
    `
    mainDiv.innerHTML = markup
    messageArea.appendChild(mainDiv)
}


//Recieve the message
socket.on('message',(msg) =>{
    appendMessage(msg,'incoming')
    scrollToBottom()
```

```
})

function scrollToBottom()
{
   messageArea.scrollTop=messageArea.scrollHeight
}
```