

**MINI PROJECT-2**  
**(2021-22)**  
**“Chat-On”**  
**Project Report**

**GLA University, Mathura**

**Submitted By -**

Aryan Jaiswal (191500158)

Divyam Goel (191500272)

**Under the Supervision Of**

**Dr. Ruchi Gupta**

**Sr. Technical Trainer**

**Department of Computer Engineering & Applications**



Department of Computer Engineering and Applications  
GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,  
Chaumuha, Mathura – 281406 U.P (India)

## **Declaration**

I/we hereby declare that the work which is being presented in the Bachelor of technology. Project “**Chat-On**”, in partial fulfillment of the requirements for the award of the ***Bachelor of Technology*** in Computer Science and Engineering and submitted to the Department of Computer Engineering and Applications of GLA University, Mathura, is an authentic record of my/our own work carried under the supervision of **Dr. Ruchi Gupta, Sr. Technical Trainer, Dept. of CEA, GLA University.**

The contents of this project report, in full or in parts, have not been submitted to any other Institute or University for the award of any degree.

**Sign:** *Aryan Jaiswal*

**Sign:** *Divyam Goel*

**Name of Candidate:** Aryan Jaiswal

**Name of Candidate:** Divyam Goel

**University Roll No.:** 191500158

**University Roll No.:** 191500272



**Department of Computer Engineering and Applications**  
**GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,**  
**Chaumuha, Mathura – 281406 U.P (India)**

## **Certificate**

This is to certify that the project entitled “Chat-On”, carried out in  
Mini Project – II Lab, is a bona fide work by Aryan Jaiswal and Divyam Goel is submitted  
in partial fulfillment of the requirements for the award of the degree Bachelor of  
Technology (Computer Science & Engineering).

**Signature of Supervisor:**

**Name of Supervisor:** Dr. Ruchi Gupta

**Date:**

# Training Certificates

- **Aryan Jaiswal**



- Divyam Goel





**Department of Computer Engineering and Applications**  
**GLA University, 17 km. Stone NH#2, Mathura-Delhi Road,**  
**Mathura – 281406 U.P (India)**

## **ACKNOWLEDGEMENT**

Presenting the ascribed project paper report in this very simple and official form, we would like to place my deep gratitude to GLA University for providing us the instructor Dr. Ruchi Gupta, our Sr. Technical Trainer, Dept. of CEA.

She has been helping us since Day 1 in this project. She provided us with the roadmap, the basic guidelines explaining on how to work on the project. She has been conducting regular meeting to check the progress of the project and providing us with the resources related to the project. Without her help, we wouldn't have been able to complete this project.

And last but not the least we would like to thank our dear parents for helping us to grab this opportunity to get trained and also my colleagues who helped us find resources during the training.

Thank You

**Sign:** Aryan Jaiswal

**Sign:** Divyam Goel

**Name of Candidate:** Aryan jaiswal

**Name of Candidate:** Divyam Goel

**University Roll No.:** 191500158

**University Roll No:** 191500272

## **ABSTRACT**

In this project, we are creating a website, which we have named “Chat-On” This website will provide us a platform to communicate with people all over the world. All the users will be having their separate accounts on the website which will be connected to their email ID. Users can send messages to individual as well as in groups ,users can search for other user using their name or username, users can start a thread in groups for a particular comment, user can send files in chats and embed YouTube videos in chat so they don’t have to leave the window to watch the video ,users can react to individual chats using emojis.

Chat-On is such an application which provide you with the tools to connect to people close to you and bring them closer.

# INTRODUCTION

## 1.1 CONTEXT

This Website “Chat-On” has been submitted in partial fulfilment of the requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering at GLA University, Mathura supervised by Dr. Ruchi Gupta. This project has been completed approximately three months and has been executed in modules, meetings have been organized to check the progress of the work and for instructions and guidelines.

## 1.2 MOTIVATION

Texting is the new normal for people as hand written letters will extinct in the later years, it is one of the biggest developments of the 21st century and has a broader scope for more development.

## 1.3 OBJECTIVE

The goal of the website was to provide a way to the people to communicate to anyone and everyone in the world. This website will provide us a platform to communicate with people all over the world. All the users will be having their separate accounts and use it to exchange messages send photos etc.

## 1.4 EXISTING SYSTEM

There are various applications like WhatsApp, Facebook, Instagram, reddit and more which allows a user to interact with other users. They can send messages, audio messages, files and with much more features. This website is in its starting phase and with limited features but people can easily do the thing which they sign up for like sending and receiving messages, emojis, files group chat etc.

## 1.5 SOURCES

The source of our project (including all the project work, documentations and presentations) is available at the following link.

<https://github.com/aryanjais/Chat-On>



# **SOFTWARE REQUIREMENT ANALYSIS**

## **2.1 IMPACT OF CHAT-ON ON DAILY LIFE**

Increasingly, people are living active digital lives. They're becoming more and more addicted to digital devices. These include various applications and services. Digital means of communication help users always stay in touch with family and friends. Also, it connects you with business colleagues and clients. This is real communication freedom. There's a basic need for keeping things small and simple. As a result, social networking sites are becoming more conversational. That's why messaging apps like Chat-On, which are practical and easy-to-use, attract more users.

## **2.2 PROBLEM STATEMENT**

Chat rooms have become a popular way to support a forum for n-way conversation or discussion among a set of people with interest in a common topic. Chat applications range from simple, text-based ones to entire virtual worlds with exotic graphics. In this project you are required to implement a simple text-based chat client/server application.

Problem description:

Email, newsgroup and messaging applications provide means for communication among people but these are one-way mechanisms and they do not provide an easy way to carry on a real-time conversation or discussion with people involved. Chat room extends the one-way messaging concept to accommodate multi-way communication among a set of people.

## 2.3 HARDWARE AND SOFTWARE REQUIREMENTS

### Hardware Requirement

- Processor : Intel i5
- Operating System : Any Operating System
- RAM : 4 GB (or higher)
- Hard disk : 256GB

### Software Requirement

- Software used: VS Code
- Language used : HTML, CSS, JavaScript, Node JS
- Database: Express
- User Interface Design : Website UI

## 2.4 MODULES AND FUNCTIONALITIES

- **Home Screen:** Home screen shows the chats, individual user, group chats, search bar.  
User can send messages, files etc
- **Login Page:** This page is for registered users who have already registered themselves on the website and have a username and a password. There is also a way on this page for the new users to register themselves which will take them to the registration page.
- **Registration Page:** This is page is solely designed for the new users of the website

who are willing to register themselves. This page takes input of the various details of the user and stores it in the database, later helping the user to login into the account with credentials they have provided.

- **Chat Screen:** This is where users will send and receive messages, media, links etc.
- **Logout page:** Then is this last panel for the users to sign out from the account. As soon as the users sign out they are brought back to the login page.

# TECHNOLOGY USED

## 4.1 WEB DEVELOPMENT

Web development refers to the building, creating, and maintaining of websites. It includes aspects such as web design, web publishing, web programming, and database management. It is the creation of an application that works over the internet i.e. websites. The word Web Development is made up of two words, that is:

- **Web:** It refers to websites, web pages or anything that works over the internet.
- **Development:** Building the application from scratch.

Web Development can be classified into two ways:

- **Frontend Development**
- **Backend Development**

**Frontend Development:** The part of a website that the user interacts directly is termed as front end. It is also referred to as the 'client side' of the application.

**Backend Development:** Backend is the server side of a website. It is the part of the website that users cannot see and interact. It is the portion of software that does not come in direct contact with the users. It is used to store and arrange data.

## 4.3 TOOLS AND LANGUAGES

- **Vscode:** Visual Studio Code is a source-code editor made by Microsoft for Windows, Linux and macOS. Features include support for debugging, syntax highlighting, intelligent code completion, snippets, code refactoring, and embedded Git. Users can change the theme, keyboard shortcuts, preferences, and install extensions that add additional functionality.

Languages used in building a website are classified as per the Front End and Back End. For designing the Front End of the website we have used CSS and for designing the Back End we have used MongoDB.

- **HTML:** The Hyper Text Markup Language, or HTML is the standard markup language for documents designed to be displayed in a web browser. It can be assisted by technologies such as Cascading Style Sheets (CSS) and scripting languages such as JavaScript. Web browsers receive HTML documents from a web server or from local storage and render the documents into multimedia web pages. HTML describes the structure of a web page semantically and originally included cues for the appearance of the document. HTML elements are the building blocks of HTML pages. With HTML constructs, images and other objects such as interactive forms may be embedded into the rendered page. HTML provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. HTML elements are delineated by tags, written using angle brackets. HTML can embed programs written in a scripting language such as JavaScript, which affects the behavior and content of web pages.
- **CSS:** CSS (Cascading Style Sheets) is a stylesheet language used to design the webpage to make it attractive. The reason of using CSS is to simplify the process of making web pages presentable. CSS allows you to apply styles to web pages. More importantly, CSS enables you to do this independent of the HTML that makes up each web page. There are

three types of CSS which are given below:

- **Inline CSS**
  - **Internal or Embedded CSS**
  - **External CSS**
- 
- **JavaScript:** JavaScript is the world most popular lightweight, interpreted compiled programming language. It is also known as scripting language for web pages. It is well-known for the development of web pages, many non-browser environments also use it. JavaScript can be used for Client-side developments as well as Server-side developments.

JavaScript can be added to your HTML file in two ways:

- **Internal JS:** We can add JavaScript directly to our HTML file by writing the code inside the `<script>` tag. The `<script>` tag can either be placed inside the `<head>` or the `<body>` tag according to the requirement.
- **External JS:** We can write JavaScript code in other file having an extension `.js` and then link this file inside the `<head>` tag of the HTML file in which we want to add this code.
- **Express:** Express.js, or simply Express, is a backend web application framework for Node.js, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It has been called the de facto standard server framework for Node.js
- **Getstream API:** This API contains React Chat Messaging Component Library which includes everything you need to build a fully functioning chat user experience in React with support for rich messages, reactions, threads, image uploads and videos

## 4.4 BASIC TERMINOLOGY

- **CSS File:** CSS (Cascading Style Sheets) are files that describe how HTML elements are displayed on the screen, paper, etc. With HTML, you can have either embedded styles or styles can be defined in an external stylesheet. For embedding the styles, the

`<style></style>` tags are used. The external stylesheets are stored in files with the .css extension. With the external CSS, you can include it on multiple HTML pages to update the style of those pages. Even a single CSS file can be used to style a complete website.

- **HTML file**: HTML is a HyperText Markup Language file format used as the basis of a web page. HTML is a file extension used interchangeably with HTM. HTML consists of tags surrounded by angle brackets. The HTML tags can be used to define headings, paragraphs, lists, links, quotes, and interactive forms. It can also be used to embed JavaScript and CSS (cascading style sheets).
- **JavaScript File**: Within a browser, JavaScript doesn't do anything by itself. You run JavaScript from inside your HTML webpages. To call JavaScript code from within HTML, you need the `<script>` element. There are two ways to use script, depending on whether you're linking to an external script or embedding a script right in your webpage
- **API**: Short for Application Programming Interface. APIs are functions that developers can call on to access specific features by calling upon programs, code, and services that others have written.

# IMPLEMENTATION AND USER INTERFACE

Creating a website concept design with screen sketches and functional flow diagrams is the best way to communicate your vision to the website developer. Making the concept clear to the developer is probably the most important factor in successful web development. Yet it is one of the most common problems or obstacles in web development outsourcing project.

No matter what the marketing and profit goals are or if you are outsourcing an app for your personal use, you need to fully design and document the website concept if you expect a programmer to make your vision a reality. Developers are not mind readers and even descriptions given during conversations can be very fleeting or interpreted differently. Fully documenting your concept, therefore, leaves little to chance. The two most important things to do are:

- A) Make a comprehensive description of how the website works and what it does (functionality)
- B) Create a comprehensive description of what the user sees and does (look and feel).

## 5.1 Implementation of Chat-On:

Implementation of the website Chat-On is taken place in various phases. Firstly we build the login interface then home page i.e. make fragment for each of the list item using the Components and then make various layout for the supporting features and connect the website with the database for fetching the login credential of the user.

### 5.1.1 Step to be followed to develop the website:

1. First we create the home page for the new user to signup.
2. After that we create login page which comprises of various phases that are mentioned below:
  - Login Page: allows user to login into the website if the user is existing one
  - Register Page: If the user is new to our website then firstly she/she have to register themselves on the app.



3. Creating Components for each of the Functionality. The components are:
  - Auth
  - Channel Container
  - Channel search
  - Edit Channels
  - Team Channel
  - Team Message
  - User List
4. Now we have created various Components like Auth, Channel Container, Channel Search and many more.
5. After that we combine the components file together into the App.js file.
6. Now we fetch the data (that we have received from getstream API) to server folder
7. In the Main Page Activity there are various functionality. Some of them are mentioned below
  - Search user/Channel: It will allow user to find other users or channels in the website.
  - Individual Messages: This section is for sending individual messages to other users.
  - Channels: This section will contain all the groups to chat from created and joined by the user.
  - Logout Button: This button is to log user out from their session.
  - Attach files: It will allow user to attach files to their chats such as photos, videos, documents etc.
  - Edit Channel: This button is used to edit the channel name and users.

# USER INTERFACE

## Sign Up

Full Name

Username

Phone Number

Avatar URL

Password

Confirm Password

[Sign Up](#)

Already have an account?[Sign In](#)

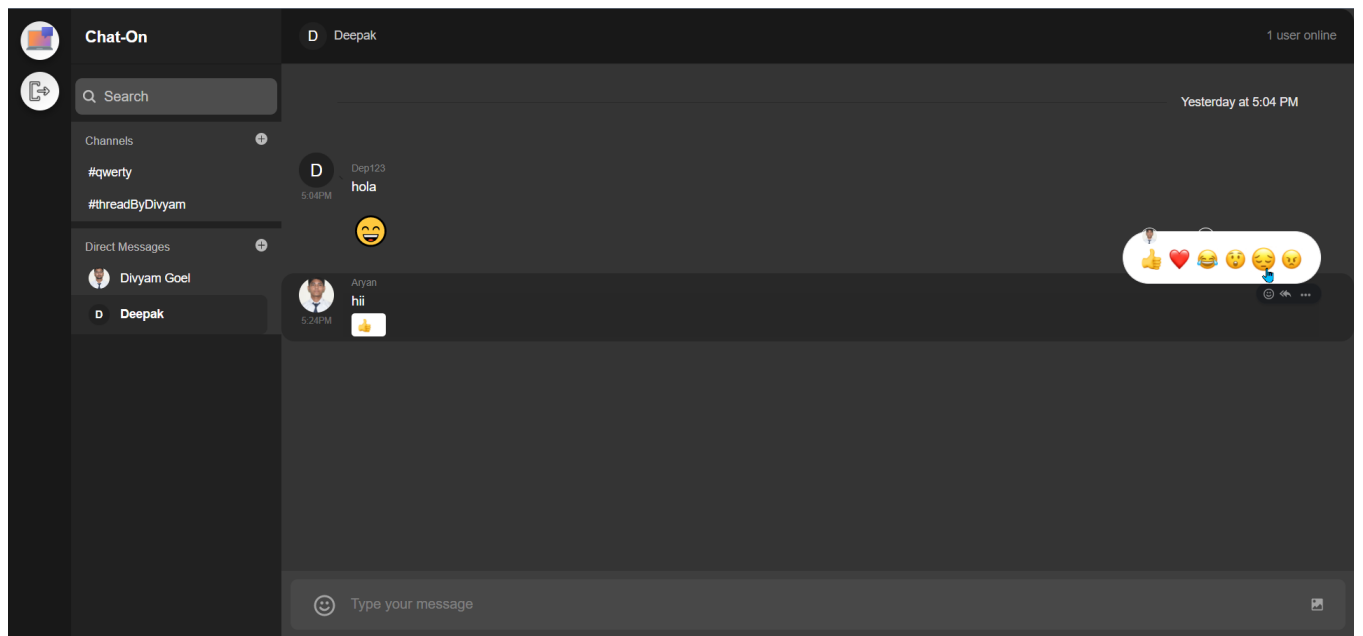
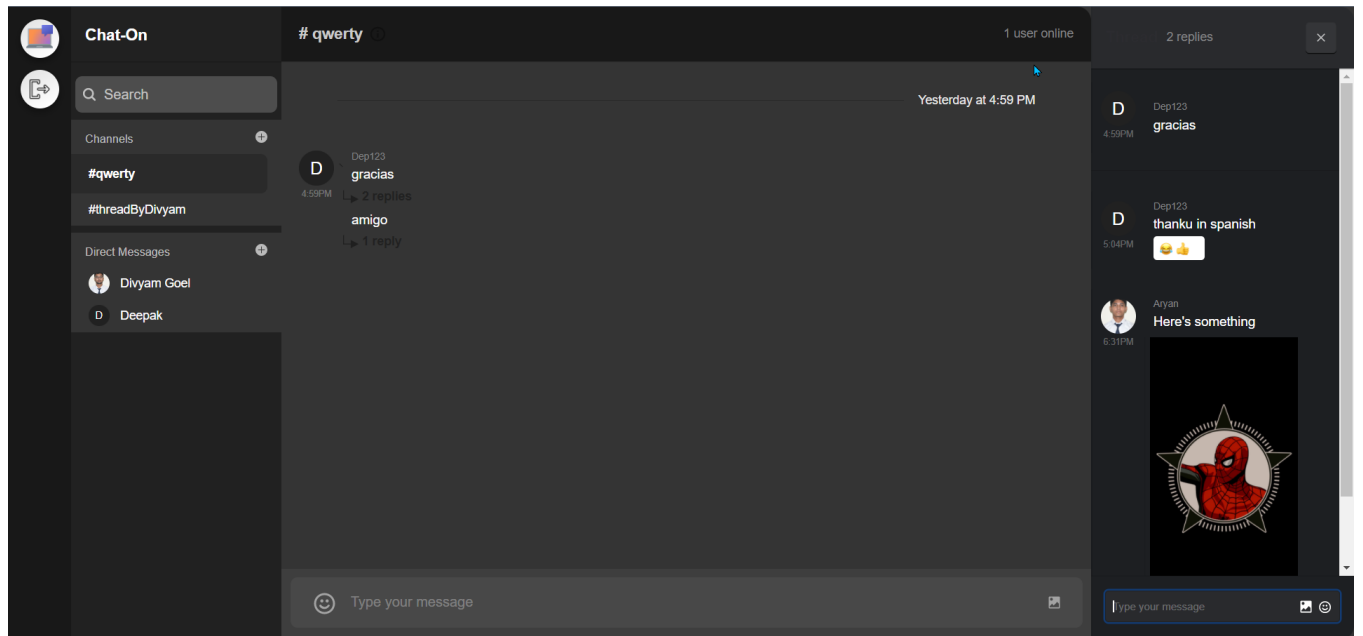
## Sign In

Username

Password

[Sign In](#)

Don't have an account?[Sign Up](#)





Chat-On

Online



Search

Channels



#qwerty

#threadByDivyam

Direct Messages



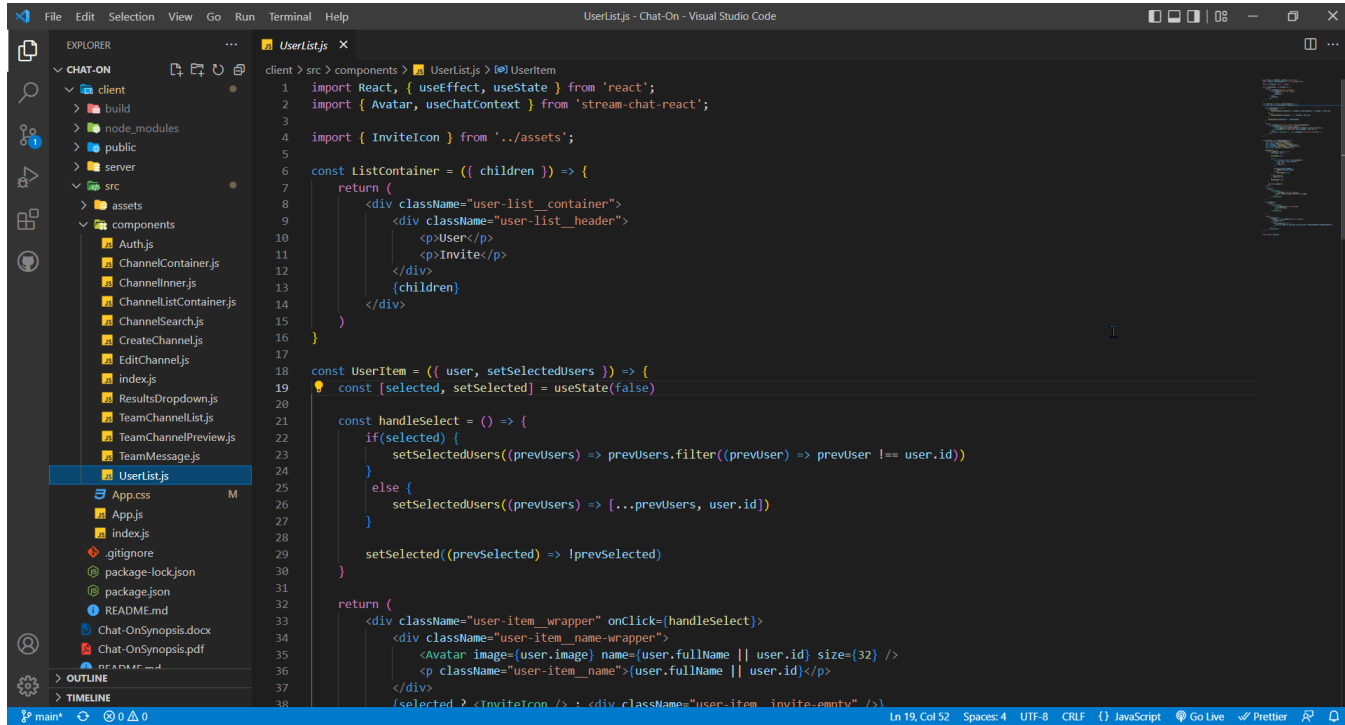
Divyam Goel



Deepak

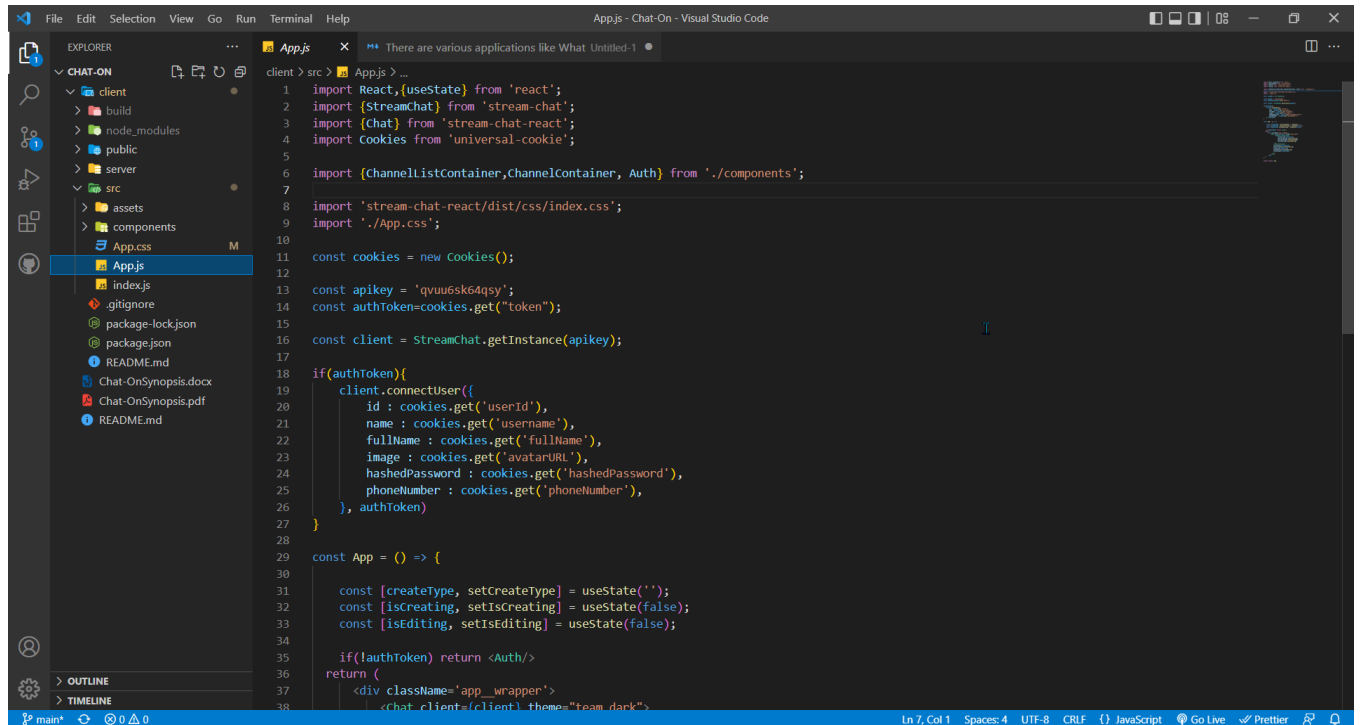


# CODE SNIPPETS



This screenshot shows the Visual Studio Code editor with the file `UserList.js` open. The Explorer sidebar on the left shows the project structure, with the `components` directory selected. The code in the editor defines a `ListContainer` component and a `UserItem` component. The `UserItem` component uses `useState` to manage the selection state and `handleSelect` to update the selected users.

```
1 import React, { useEffect, useState } from 'react';
2 import { Avatar, useChatContext } from 'stream-chat-react';
3
4 import { InviteIcon } from '../assets';
5
6 const ListContainer = ({ children }) => {
7   return (
8     <div className="user-list_container">
9       <div className="user-list_header">
10        <p>User</p>
11        <p>Invite</p>
12      </div>
13      {children}
14    </div>
15  )
16 }
17
18 const UserItem = ({ user, setSelectedUsers }) => {
19   const [selected, setSelected] = useState(false)
20
21   const handleSelect = () => {
22     if(selected) {
23       setSelectedUsers((prevUsers) => prevUsers.filter((prevUser) => prevUser !== user.id))
24     } else {
25       setSelectedUsers((prevUsers) => [...prevUsers, user.id])
26     }
27   }
28
29   setSelected((prevSelected) => !prevSelected)
30 }
31
32 return (
33   <div className="user-item_wrapper" onClick={handleSelect}>
34     <div className="user-item_name-wrapper">
35       <Avatar image={user.image} name={user.fullName || user.id} size={32} />
36       <p className="user-item_name">{user.fullName || user.id}</p>
37     </div>
38     {selected ? <InviteIcon /> : <div className="user-item_invite-empty" />}
```

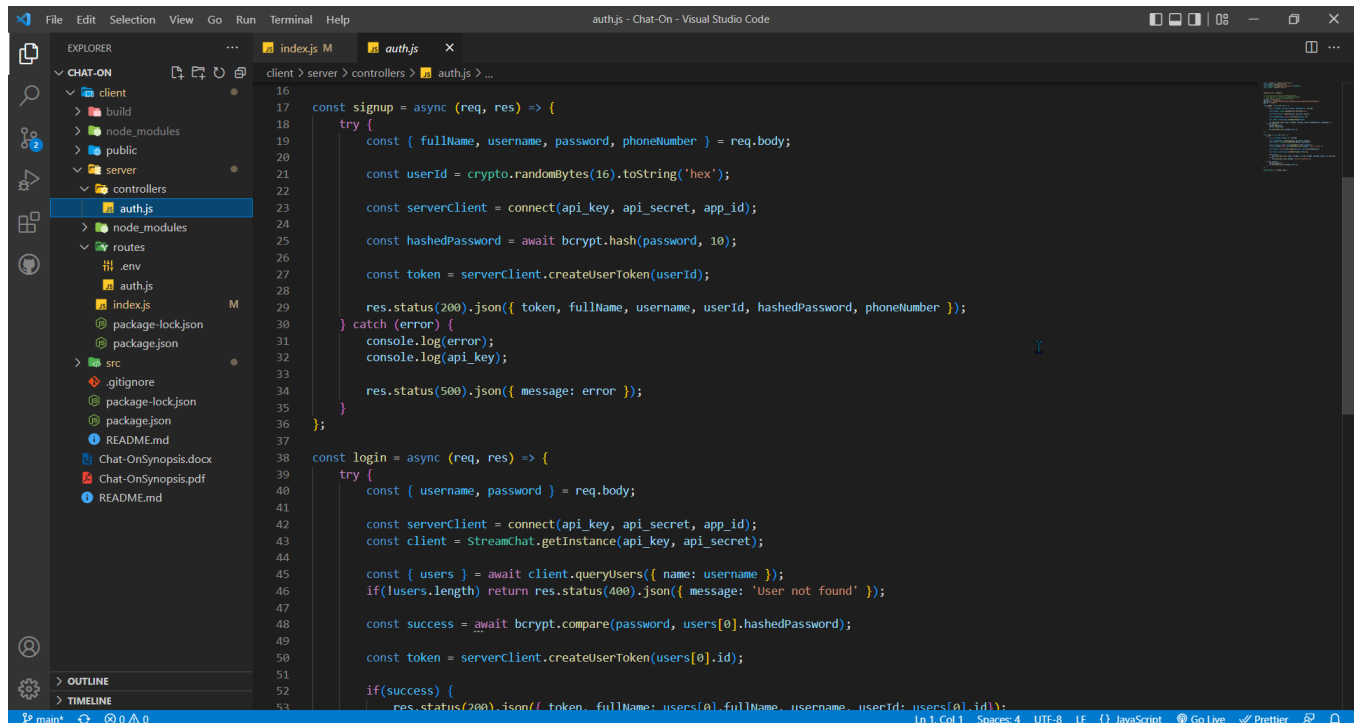


This screenshot shows the Visual Studio Code editor with the file `App.js` open. The Explorer sidebar on the left shows the project structure, with the `App.js` file selected. The code in the editor defines the `App` component, which uses `useState` to manage the create type, creating state, and editing state. It also uses `cookies` to manage user authentication.

```
1 import React, { useState } from 'react';
2 import { StreamChat } from 'stream-chat';
3 import { Chat } from 'stream-chat-react';
4 import Cookies from 'universal-cookie';
5
6 import { ChannelListContainer, ChannelContainer, Auth } from './components';
7
8 import 'stream-chat-react/dist/css/index.css';
9 import './App.css';
10
11 const cookies = new Cookies();
12
13 const apiKey = 'qvuus6sk64qsy';
14 const authToken = cookies.get("token");
15
16 const client = StreamChat.getInstance(apiKey);
17
18 if(authToken){
19   client.connectUser({
20     id : cookies.get('userId'),
21     name : cookies.get('username'),
22     fullName : cookies.get('fullName'),
23     image : cookies.get('avatarURL'),
24     hashedPassword : cookies.get('hashedPassword'),
25     phoneNumber : cookies.get('phoneNumber'),
26   }, authToken)
27 }
28
29 const App = () => {
30
31   const [createType, setCreateType] = useState('');
32   const [isCreating, setIsCreating] = useState(false);
33   const [isEditing, setIsEditing] = useState(false);
34
35   if(!authToken) return <Auth/>
36
37   return (
38     <div className="app_wrapper">
39       <chat clients={client} theme="team-dark">
```

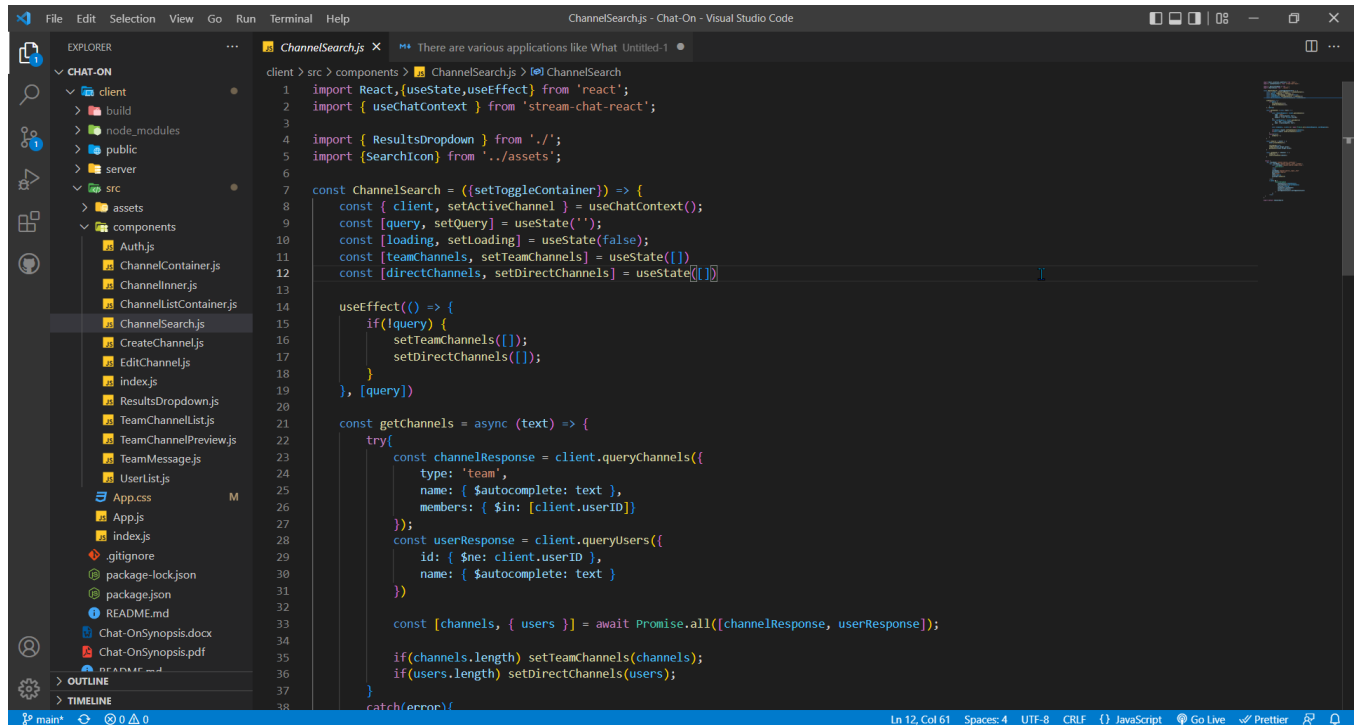
```
client > src > components > TeamChannelList.js > TeamChannelList
1 import React from 'react'
2 import { AddChannel } from '../assets';
3
4 const TeamChannelList = ({setToggleContainer, children, error = false, loading, type, isCreating, setCreateType, setIsCreating, setIsEditing}) => {
5   if(error){
6     return type === 'team'? (
7       <div className='team-channel-list'>
8         <p className='team-channel-list_message'>
9           Connection error, please wait a moment and try again.
10         </p>
11       </div>
12     ):null
13   }
14
15   if (loading){
16     return (
17       <div className='team-channel-list'>
18         <p className='team-channel-list_message loading'>
19           {type === 'team' ? 'Channels' : 'Messages'} loading...
20         </p>
21       </div>
22     )
23   }
24   return (
25     <div className='team-channel-list'>
26       <div className='team-channel-list_header'>
27         <p className='team-channel-list_header_title'>
28           {type === 'team' ? 'Channels' : 'Direct Messages'}
29         </p>
30         <AddChannel
31           isCreating={isCreating}
32           setIsCreating={setIsCreating}
33           setCreateType={setCreateType}
34           setIsEditing={setIsEditing}
35           type={type === 'team'? 'team': 'messaging'}
36           setToggleContainer={setToggleContainer}
37         />
38     </div>
39   )
40 }
```

```
client > src > components > ChannelContainer.js > ChannelContainer
1 import React from "react";
2 import { Channel, useChatContext, MessageTeam } from 'stream-chat-react';
3
4 import { ChannelInner, CreateChannel, EditChannel } from './';
5
6
7 const ChannelContainer = ({ isCreating, setIsCreating, isEditing, setIsEditing, createType }) => {
8   const {channel} = useChatContext();
9   if(isCreating) {
10     return (
11       <div className="channel_container">
12         <CreateChannel createType={createType} setIsCreating={setIsCreating} />
13       </div>
14     )
15   }
16
17   if(isEditing) {
18     return (
19       <div className="channel_container">
20         <EditChannel setIsEditing={setIsEditing} />
21       </div>
22     )
23   }
24
25   const EmptyState = () => (
26     <div className="channel-empty_container">
27       <p className="channel-empty_first">This is the beginning of your chat history.</p>
28       <p className="channel-empty_second">Send messages, attachments, links, emojis, and more!</p>
29     </div>
30   )
31
32   return (
33     <div className="channel_container">
34       <Channel
35         EmptyStateIndicator={EmptyState}
36         Message={messageProps => <MessageTeam key={i} {...messageProps} />
37       />
38     </div>
39   )
40 }
```



Visual Studio Code editor showing the `auth.js` file in the `controllers` directory. The code defines two functions: `signup` and `login`.

```
16 const signup = async (req, res) => {
17   try {
18     const { fullName, username, password, phoneNumber } = req.body;
19
20     const userId = crypto.randomBytes(16).toString('hex');
21
22     const serverClient = connect(api_key, api_secret, app_id);
23
24     const hashedPassword = await bcrypt.hash(password, 10);
25
26     const token = serverClient.createUserToken(userId);
27
28     res.status(200).json({ token, fullName, username, userId, hashedPassword, phoneNumber });
29   } catch (error) {
30     console.log(error);
31     console.log(api_key);
32
33     res.status(500).json({ message: error });
34   }
35 }
36
37
38 const login = async (req, res) => {
39   try {
40     const { username, password } = req.body;
41
42     const serverClient = connect(api_key, api_secret, app_id);
43     const client = StreamChat.getInstance(api_key, api_secret);
44
45     const { users } = await client.queryUsers({ name: username });
46     if (users.length) return res.status(400).json({ message: 'User not found' });
47
48     const success = await bcrypt.compare(password, users[0].hashedPassword);
49
50     const token = serverClient.createUserToken(users[0].id);
51
52     if (success) {
53       res.status(200).json({ token, fullName: users[0].fullName, username: users[0].id });
54     }
55   }
56 }
```



Visual Studio Code editor showing the `ChannelSearch.js` file in the `components` directory. The code defines a `ChannelSearch` component using `React` and `StreamChat`.

```
1 import React, { useState, useEffect } from 'react';
2 import { useChatContext } from 'stream-chat-react';
3
4 import { ResultsDropdown } from './ResultsDropdown';
5 import { SearchIcon } from './assets';
6
7 const ChannelSearch = ({ setToggleContainer }) => {
8   const { client, setActiveChannel } = useChatContext();
9   const [query, setQuery] = useState('');
10  const [loading, setLoading] = useState(false);
11  const [teamChannels, setTeamChannels] = useState([]);
12  const [directChannels, setDirectChannels] = useState([]);
13
14  useEffect(() => {
15    if (!query) {
16      setTeamChannels([]);
17      setDirectChannels([]);
18    }
19  }, [query]);
20
21  const getChannels = async (text) => {
22    try {
23      const channelResponse = client.queryChannels({
24        type: 'team',
25        name: { $autocomplete: text },
26        members: { $in: [client.userID] }
27      });
28      const userResponse = client.queryUsers({
29        id: { $ne: client.userID },
30        name: { $autocomplete: text }
31      });
32
33      const [channels, { users }] = await Promise.all([channelResponse, userResponse]);
34
35      if (channels.length) setTeamChannels(channels);
36      if (users.length) setDirectChannels(users);
37    } catch (error) {
38      // Handle error
39    }
40  }
41 }
```

# CHAPTER - 6

## TESTING

Once source code has been generated, software must be tested to uncover as many errors as possible before delivery. It is very important to work the system successfully and achieve high quality of software. Testing include designing a series of test cases that have a high likelihood of finding errors by applying software-testing techniques.

System testing makes logical assumptions that if all the parts of the system are correct, the goal will be successfully achieved. The system should be checked logically. Validations and cross checks should be there. Avoid duplications of record that cause redundancy of data.

In other Words, Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. It is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

There are different types of testing some of them are listed below:

### 6.1 Unit Testing

It focuses on smallest unit of software design. In this we test an individual unit or groups of inter related units. It is often done by programmer by using sample input and observing its corresponding outputs. In this testing technique we are primarily focuses on

- Loop methods and function is working fine or not. □ Misunderstood or incorrect Arithmetic precedence
- Incorrect Initialization

**Unit Testing of the website:**



## **Table 1: Unit Testing of My Library Management**

### **6.3 User Testing**

User testing is the process through which the interface and functions of a website, app, product, or service are tested by real users who perform specific tasks in realistic conditions. The purpose of this process is to evaluate the usability of that website or app and to decide whether the product is ready to be launched for real users.

This website was tested by our team mates and friends who are using different Laptops also tested on different emulator to check its performance and it seems to be working fine and users of this website are satisfied with the facilities and performance of the website and like the way how the website is worked.

### **6.4 Performance Testing**

In this type of testing we have checked the performances of our application under some peculiar conditions are checked. Those conditions include:

- Low memory in the device.
- The battery in extremely at a low level.
- Poor/Bad network reception.

Performance is basically tested from 2 ends, application end, and the application server end. Our app is also performing well in this phase of testing as well. And we are getting positive feedback from user of our website.

### **6.5 Compatibility Testing**

This application was tested and used on different devices. The website worked fine and is stable.

The website worked fine in portrait mode and there isn't any problem with compatibility.

On all types of testing (that we have performed above) our performing well on our website i.e.

My Library Management.

## **CHAPTER -7**

### **CONCLUSION**

The Main objective of the project is to develop a secure Chat website which is achieved through stream API. The portability of the application is achieved by the latest React technology and stream API which contains pre-existing features for chat application as a result the product has been successfully developed in terms of extendibility, portability, and maintainability and tested in order to meet all requirements that are Authentication, Integrity, Confidentiality which are specified as the four basic concepts for the secure communication over a network.

This application has wide range of scope in the upcoming era. It is impossible to arrange the hard copies of every Books so this type of application can reduce the barrier to get knowledge at any place in a cost effective, productive way. Their own personal Books list. Even individual book stores can have this system of book web promoting their brand name as Digital Marketing and can gain number of customers.