Brian Wei and Sagarkumar Patel

User Guide Documentation

SecureChat is a chat system made for remote colleagues of the company SecureTech Solutions. It uses WebSockets and is secure, convenient, and efficient for messaging.

Overview

This guide provides step-by-step instructions on how to use our Websocket chat system, https://chat-455.web.app/.

We hosted our system using Google Cloud (back-end) and Firebase (front-end). Files for file sharing are hosted on Google Cloud. Additionally, we have an Uptime Robot that monitors our backend URL. The monitoring page is: https://stats.uptimerobot.com/AQxwnAV9HS/800462842. The robot is able to detect our server, but the server does not have any resources to respond to the ping.

Prerequisites

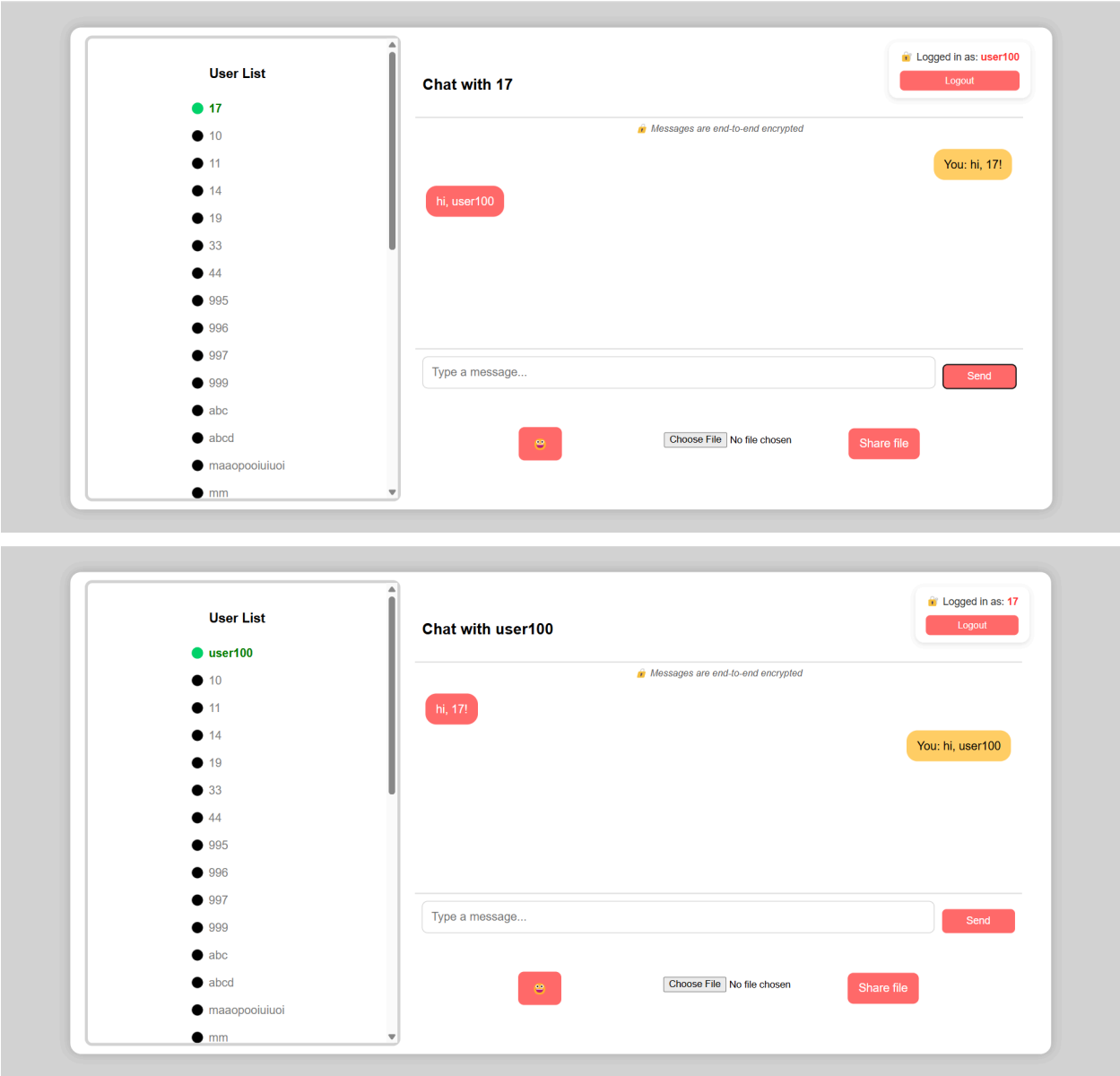
Before starting, ensure you have the following installed on your system:

- **https://chat-455.web.app/ (This is the URL to our chat.)**

Features:

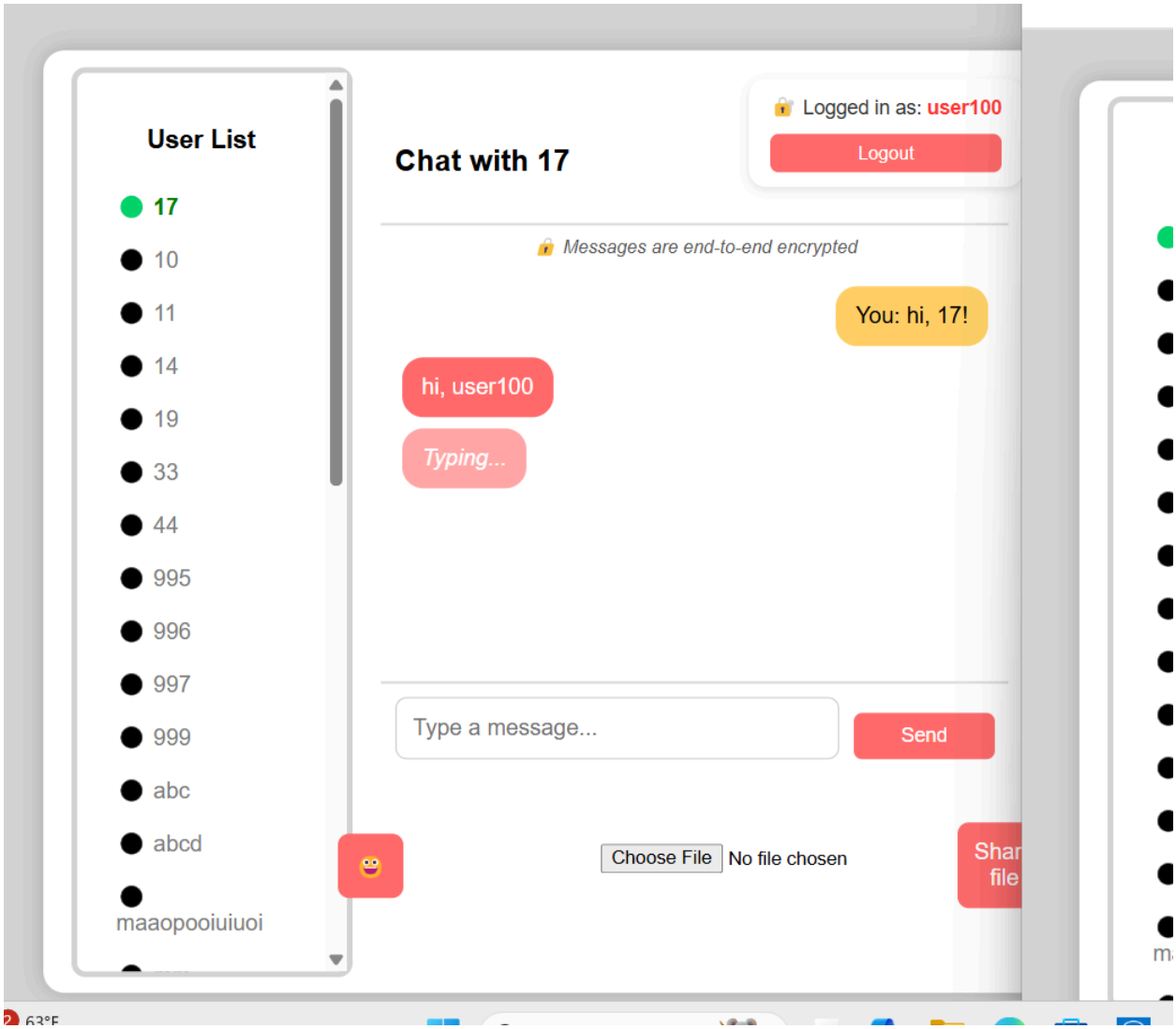
1. Chat and Status Indicator:

After logging in, you will see a list of registered users. Offline users are indicated with black dot. Online users are indicated with a green dot. Choose the online user you want to chat with. Write your message in the text box and press the button to the right of it to send. The user's message should appear for the other user automatically.



The conversation between the users, user100 and 17.

Additionally, a “typing…” indicator appears when the other user is typing.



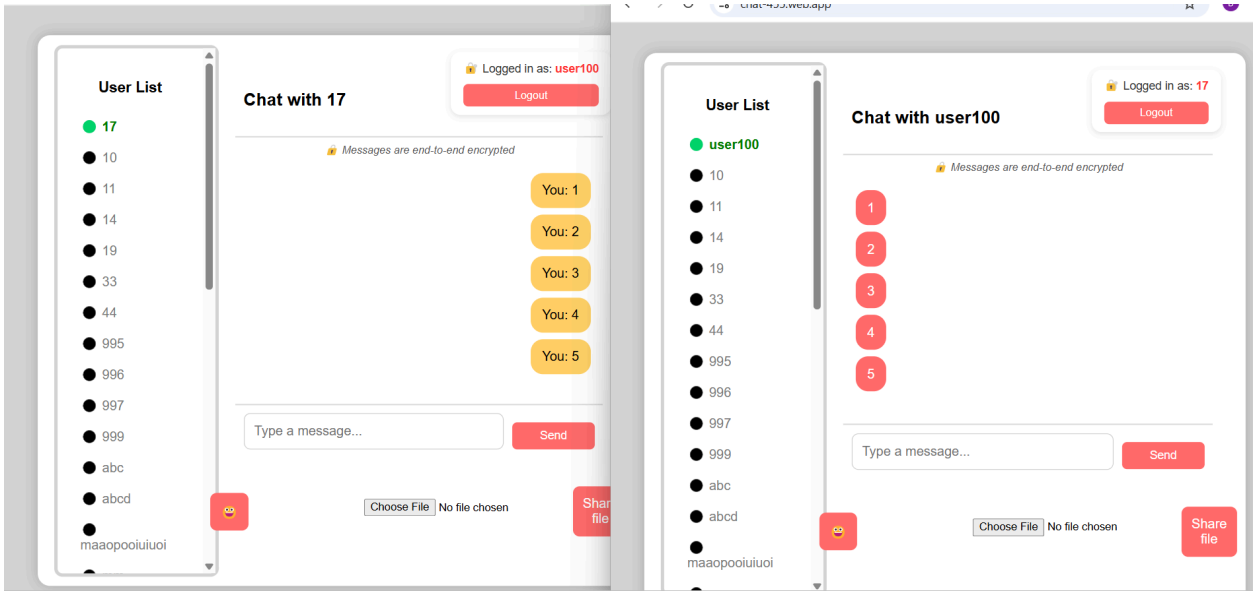
2. Test rate limiting:

Users are only allowed to send five messages within a twenty-second period. Sending more messages within a twenty-second interval will result in your message not going through to the other user. Look at the other user's window to see if the "spamming" messages came through. Because of rate limiting, they are not expected to reach the other users.

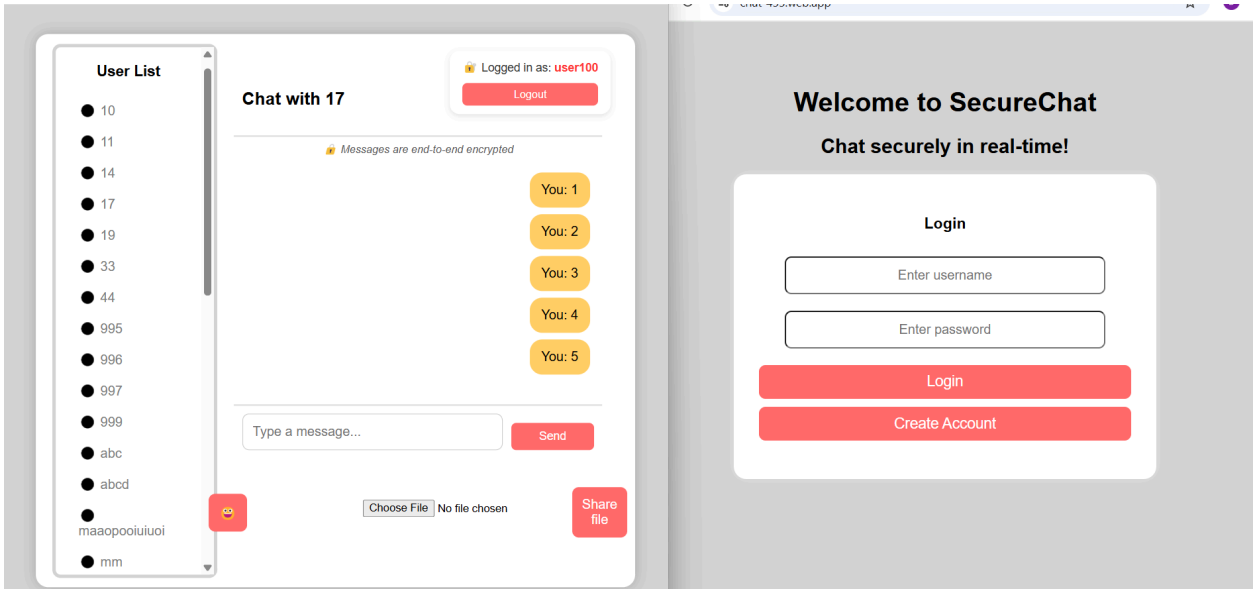
Brian Wei and Sagarkumar Patel

3. Disconnection detection:

Press the logout button for one of the users. Go to the other user that is still connected. You will see on the screen of the still-logged in user that the dot near the disconnected user is not black, meaning that the user is offline and disconnected.



User100 and 17 are connected to each other. 17's status indicator on User100's side shows he is online. User100's status indicator on 17's side shows that User100 is online too.



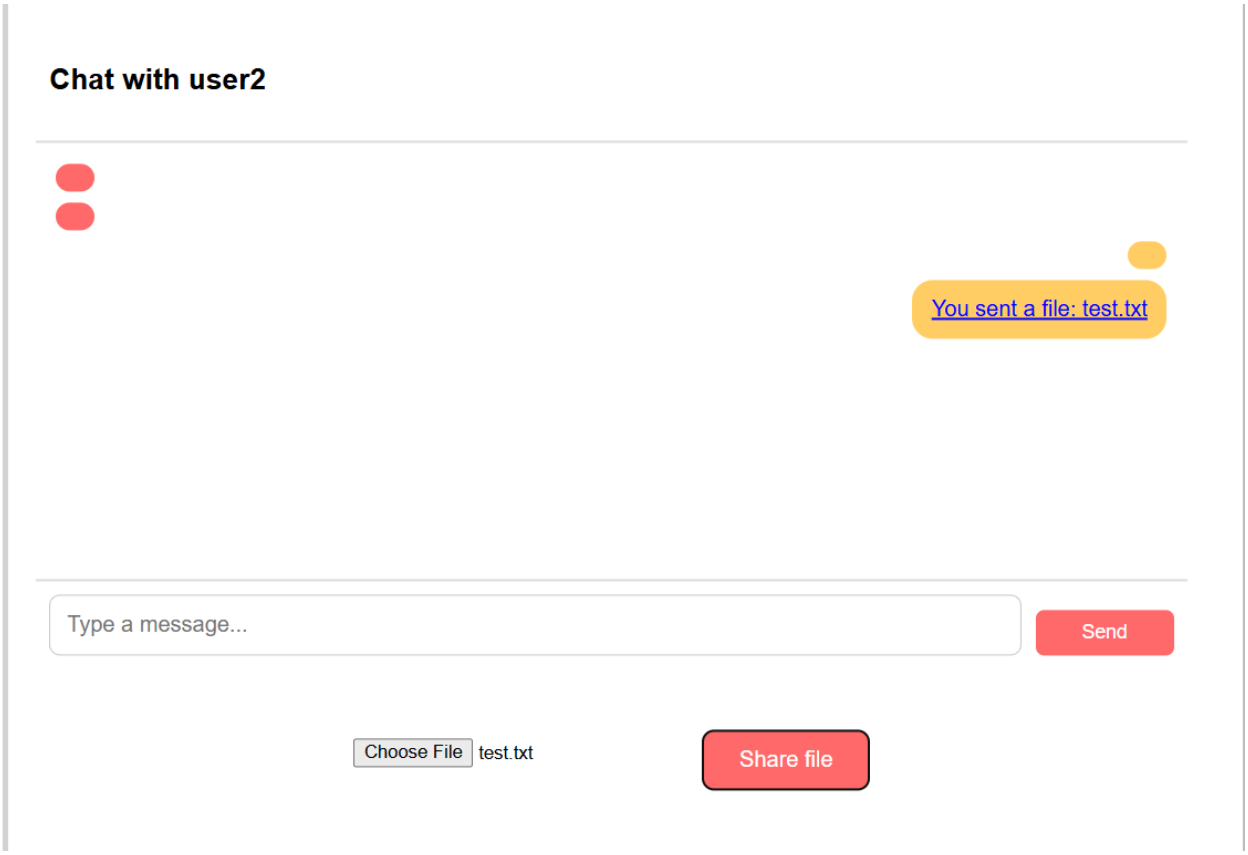
17 logs out and on User100's side, 17 is automatically marked as disconnected with the black dot.

4. **Automatic reconnection after interruption:**

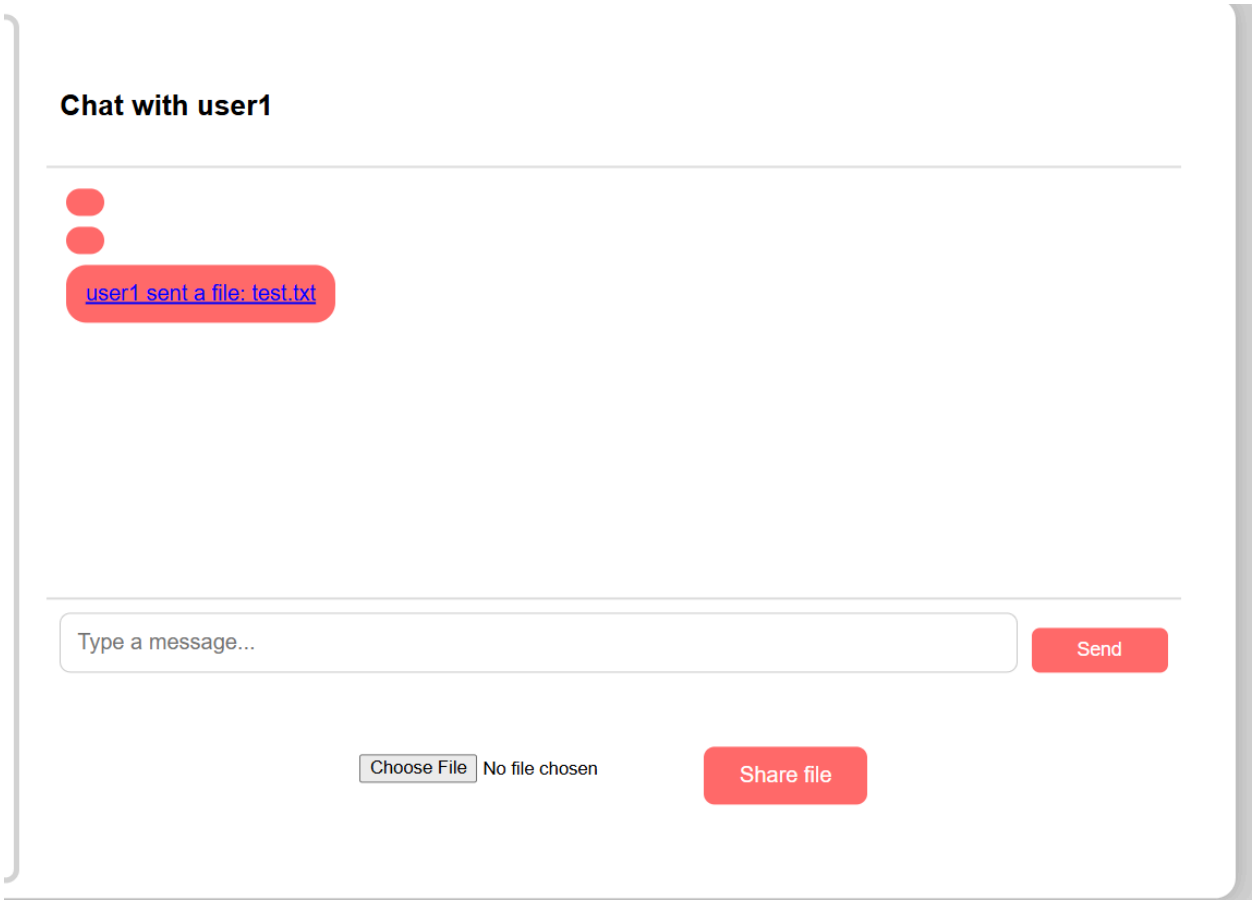
Log one user out to simulate interruption. Then log the user back in. The messages that were exchanged before the interruption occurred automatically appear. You can then send messages by clicking on the user you were chatting with. The other user does not have to do anything to re-connect.

5. **File Sharing Functionality:**

Our WebSockets system allows file sharing with encryption. After logging in, press “Choose file”. You will then select a file from your computer. Then, press share file. The file will appear on the sender’s and receiver’s chat box with a downloadable link.



The sender sends a txt file. It appears in the sender’s chat box.



The receiver receives the downloadable file link.

The above options only work for small files. For larger files, we have hosted five files in Google Cloud that you can use as a public file link. They are:

- Mars: https://storage.googleapis.com/securechat-filecloud-hosting/mars.jpg
- Meme: https://storage.googleapis.com/securechat-filecloud-hosting/meme.jpg
- Mountains: https://storage.googleapis.com/securechat-filecloud-hosting/moutains.jpg
- Squid: https://storage.googleapis.com/securechat-filecloud-hosting/squid.jpg
- Statue of Liberty: https://storage.googleapis.com/securechat-filecloud-hosting/statue.jpg

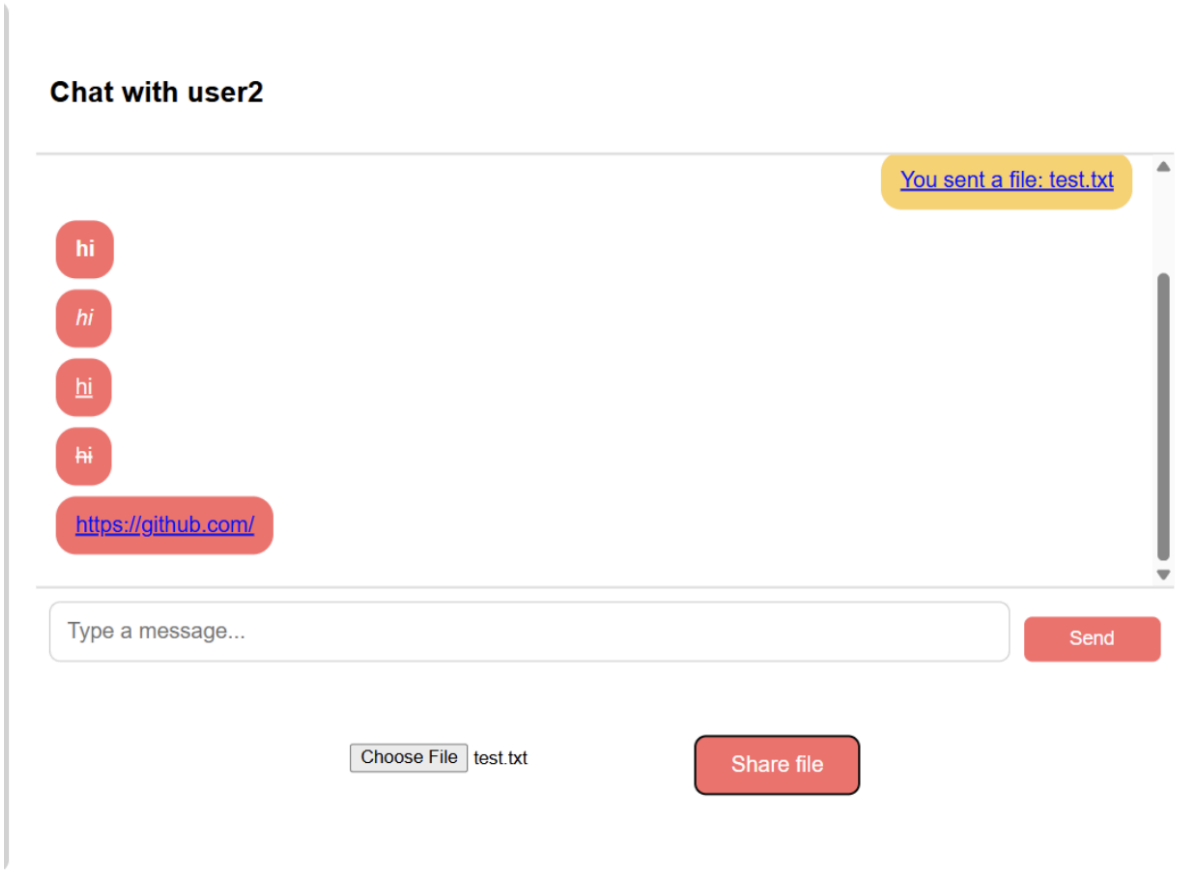
All you need to do is copy and paste these file links into the message input field and send.

Because we want to block malicious files, we have limited file sharing to files that are maximum 15 MB. The files cannot be script or exe files.

6. Basic Text Formatting

- To send bold text, press Ctrl + b before typing.
- To send italicized text, press Ctrl + i before typing.
- To send underlined text, press Ctrl + u before typing.
- To send strikethrough text, press Ctrl + x before typing.

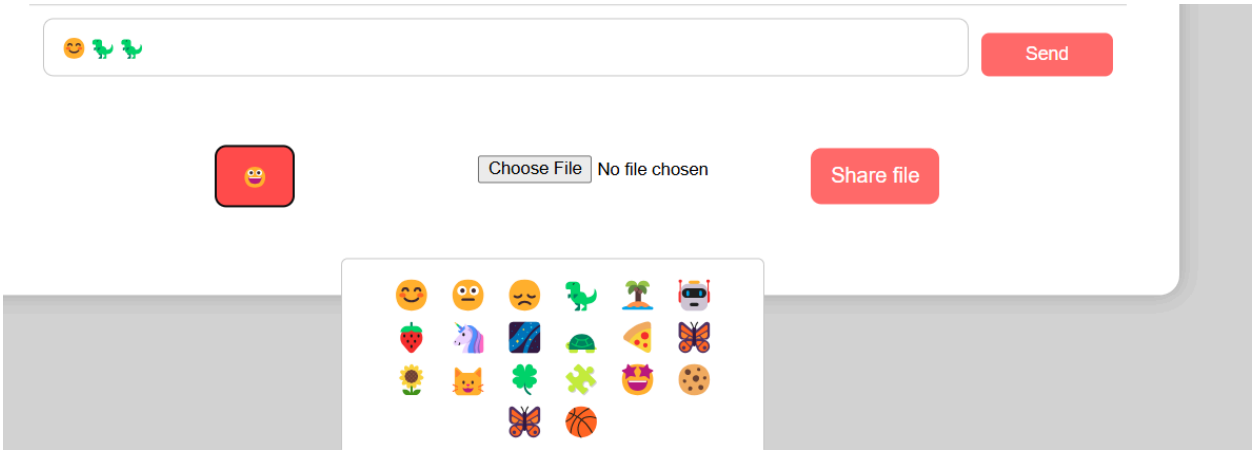
- Sending links will automatically appear as a clickable link on both sides.



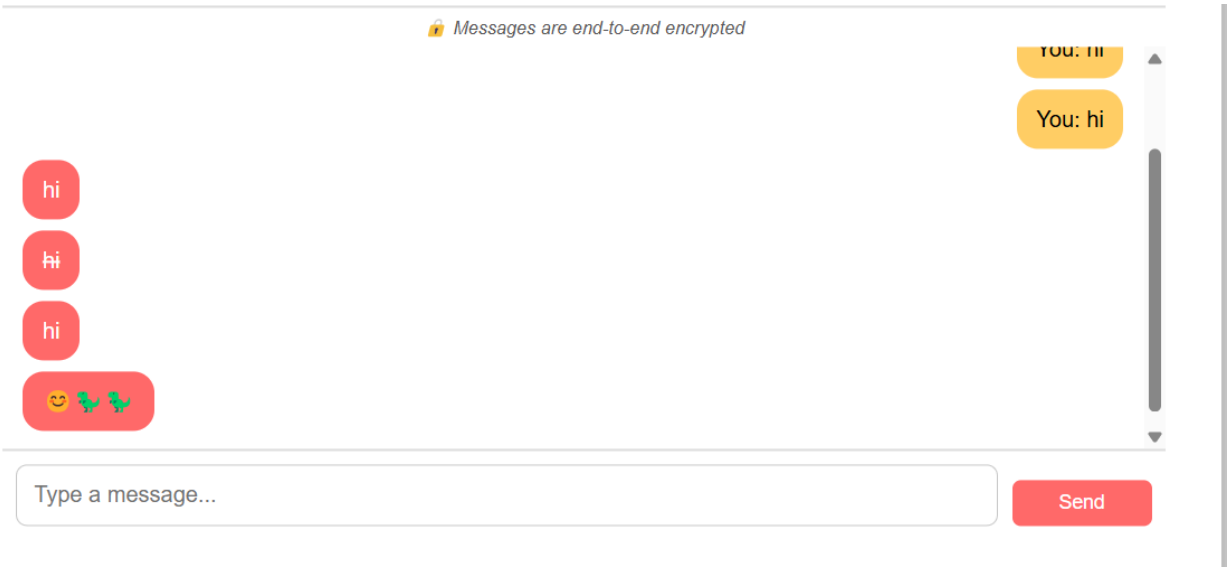
The receiver successfully receives the bold, italicized, underlined, and strikethrough text. The clickable URL link is also received successfully.

7. Unicode Emoji Picker

The chat system gives you an option to choose emojis to send. Press the 😀 button. You will see a selection of emojis you can choose from. Clicking on the chosen emoji automatically brings it to the message input field. Press send to send the emoji.



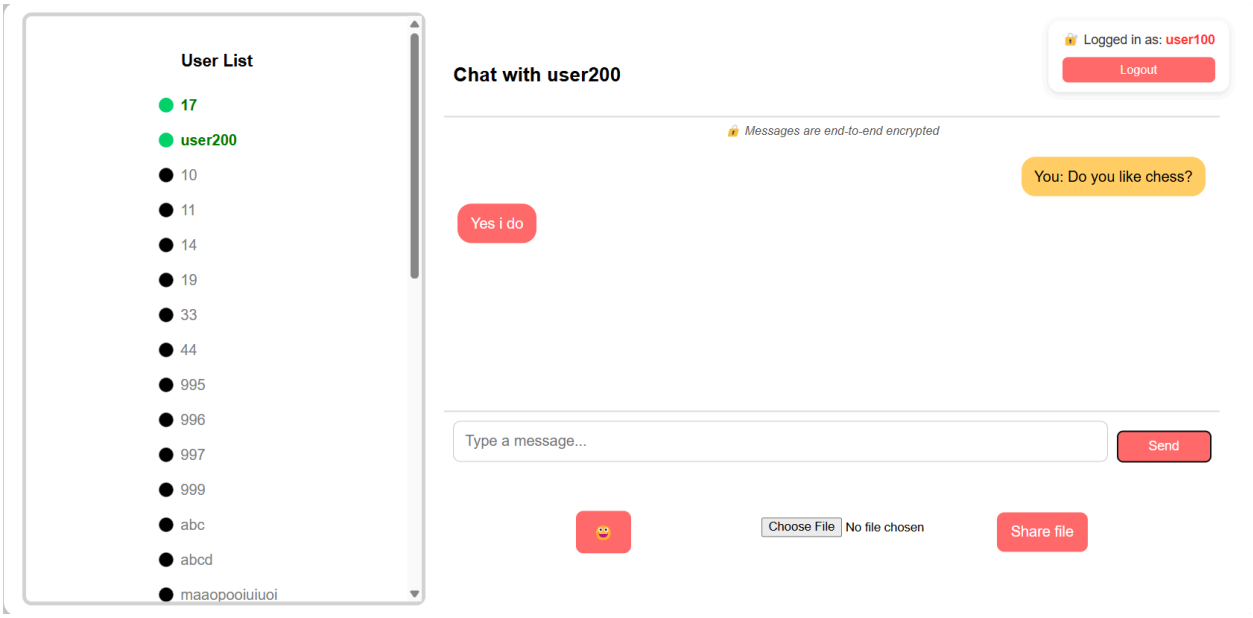
Emoji picker appears when you click the 😀 button. Clicking on the emojis causes it to immediately appear in the message input box above the emoji button.



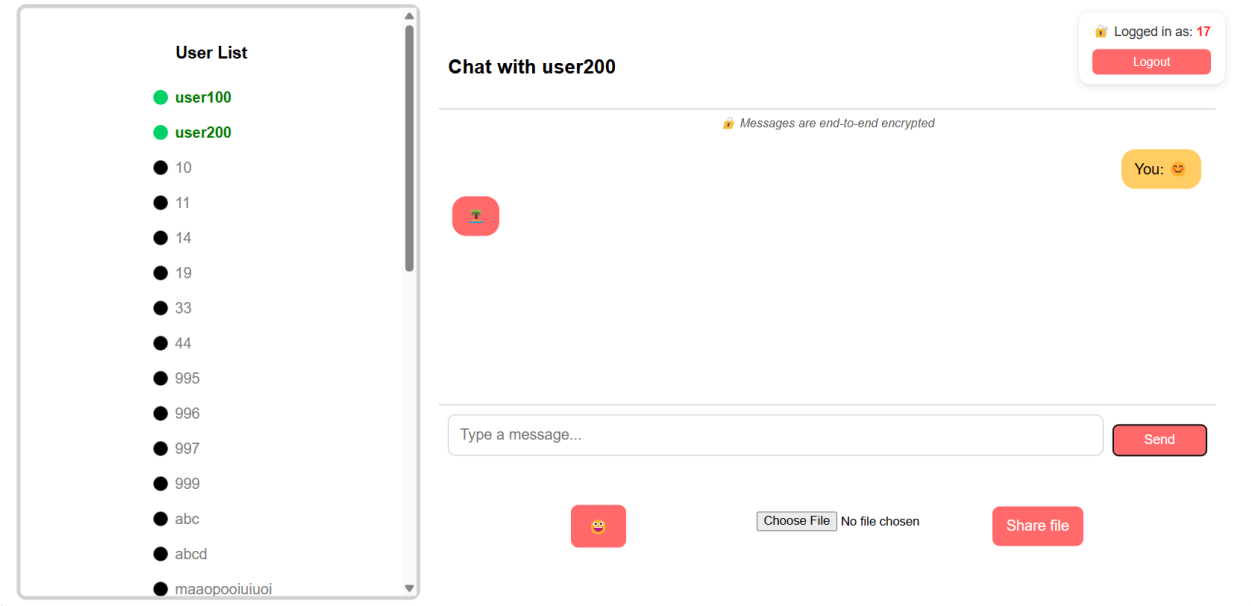
After pressing “Send”, the receiver successfully receives the emojis.

8. Multiple Users

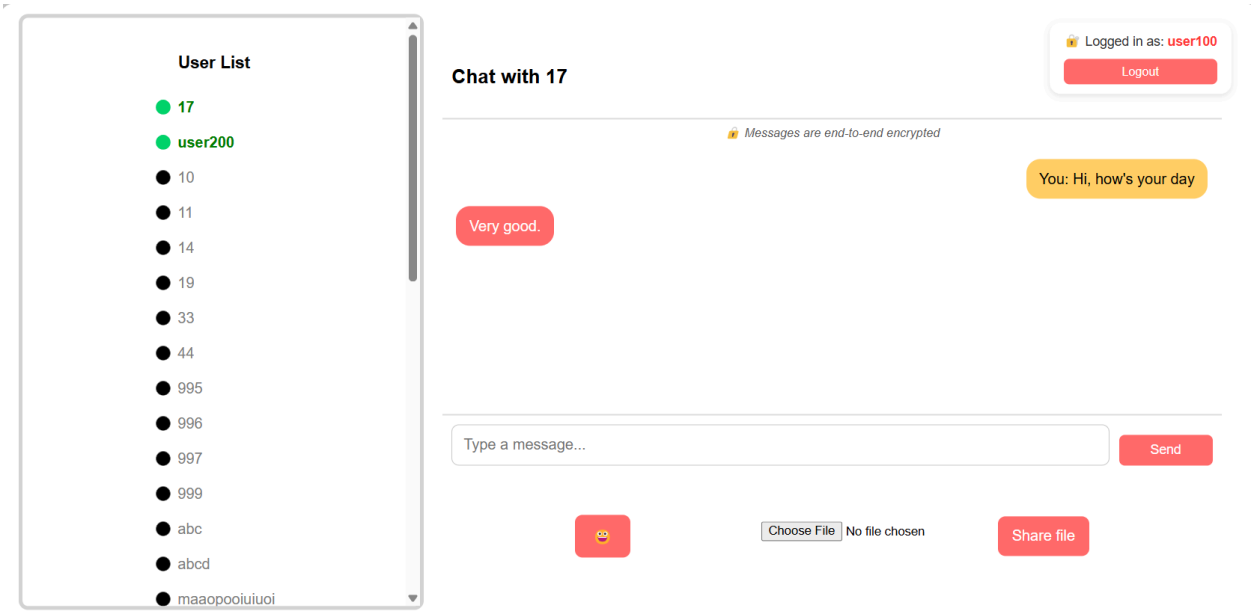
Our chat application includes the ability to have more than two users. You can select the user you want to talk to on the left panel.



The text messages between user200 and user100. The left panel allows you to easily switch between different users.



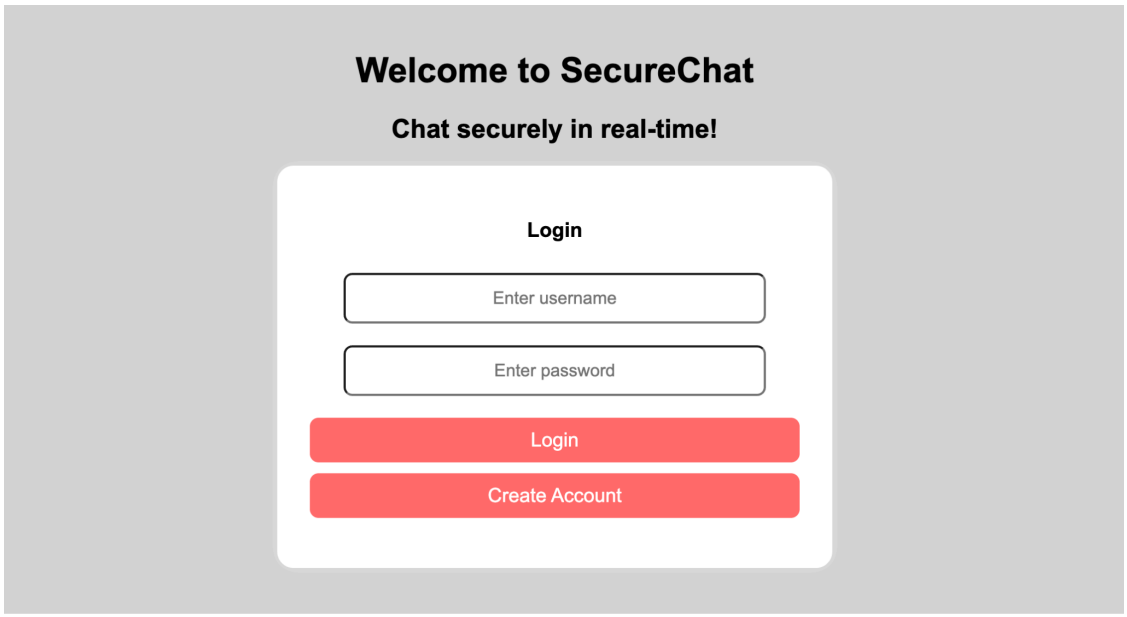
Text messages between user200 and 17.



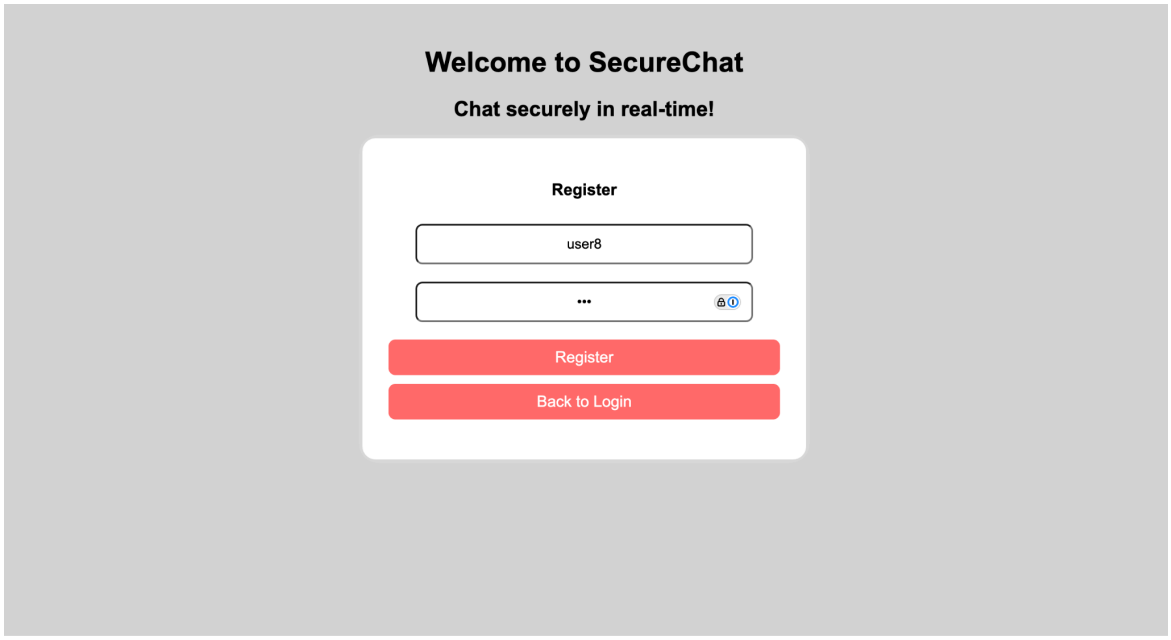
Text messages between 17 and user100.

9. User Registration/Login

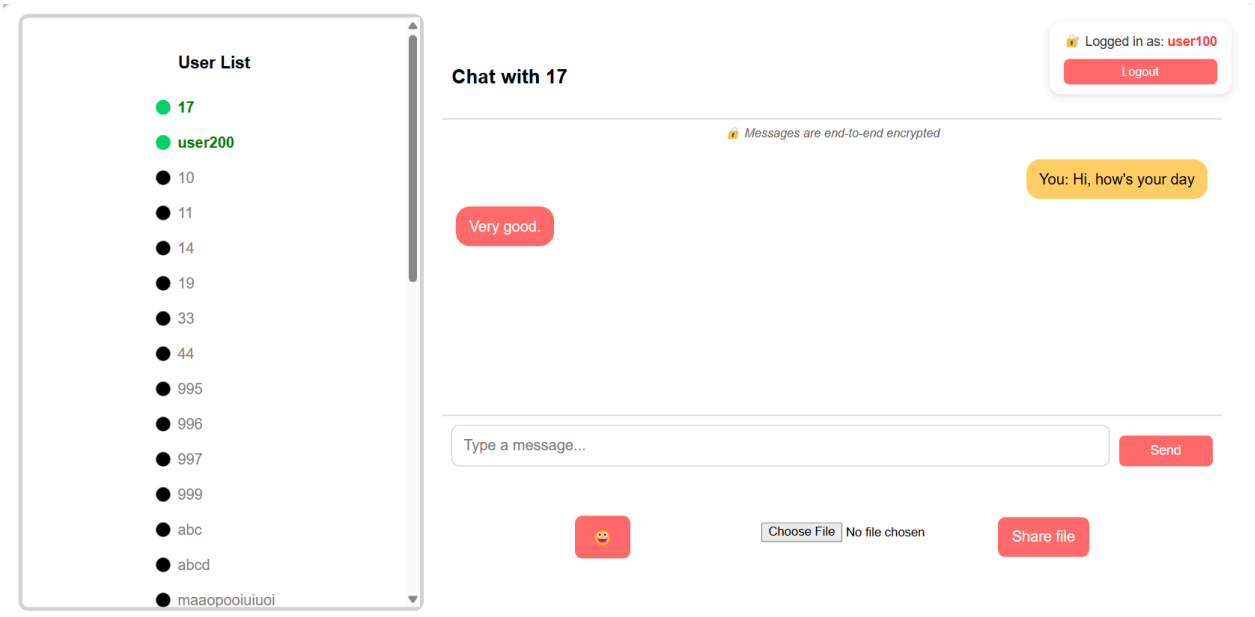
Our new update includes a feature that allows user to register their accounts alongside login with us by using our website.



Users can simply go to create an account.



Register their account, ex:user8



Users will see this screen after login is successful, which also shows all registered users on the left-hand side.

The system also has the ability to secure all passwords. Our website hashes all passwords and stores them as hashes. When users log in, the website compares the hashed passwords on the server side to ensure a seamless and secure login experience.

```
user8 $2b$12$qSdeyviZuwm9jwZOiIxAiu2hC6VVpP/YZRHJ4AV9NXku/qLJpnuFO
user6 $2b$12$4fUgBmFw2/kkr7ddSrX98uom/5Q1Q4mRGREZimn6ETeBBdNeyCNUa
abc $2b$12$VHwT8WjhyDspulgEjrmUNefr9/jj5VnhcGmKdeeX3pAFLyjekyya2
abcd $2b$12$RtsEZaZYxAhP.48JEl/P2uAyxPwlPutLC3LsEWiSzD2aKfjN4l66S
user7 $2b$12$9mT8H0PJgrE.g.xFgY0Kg.7PAUhQGtrJAwLa9mx6aDNqGsVxZ8WEC
user9 $2b$12$8daFWbrrMMeUP.I67n5.YOYhrbQ1CKtGBNqsFm4DLiQia5xyZ0yzi
user10 $2b$12$vFLpqhmb.zlIqCbXDT2kYuuWfIKd/0OJ4fpSTY881Ql8kxBXp/E.C
user11 $2b$12$kE3t1Oad8OWStOBkfmbgLufLH7kDCq0VK31/CbHv7lOGV3DJsf8fq
user50 $2b$12$gz85Sng8cA5aG18jrkEwauTB/CUcNWs6Hst2708cOUKLLM1h.DYGm
user51 $2b$12$O3xix5RW.dxHaqxhOQs5euL/i79.vXKpDqL4a.EaQdQCk1eY77uEG
user54 $2b$12$3pnYUVAr86t4iE44B9C5..EWFEsosn4qV8N9/g4QpiiWx0oPqVMbW
user57 $2b$12$iPMf271FN/O9cWSl8JKyVeJxH0cpv7XVMWjrA6Tmuy0ueOS/2/Ju6
user60 $2b$12$ZGhdMwi/69vPrkvMyo.r4O7eID5vUnRjpjYgXAQ6cHVxlf0XbU/8m
17 $2b$12$98fssEtdWCMgQBZHXusZCOJJrYu2aoqTVtFP8kIJ7Dj9oLvQF73fq
14 $2b$12$s16IIREFpbro1rAmtbZfMe7bkcmkOm3i.glq7mXgHhQO/kU33/z5O
19 $2b$12$R3a7AKwDlW8SyW0pAZ/FSeOWaK5Cew1UxIbxZuwbsmFTI8frZFVWa
10 $2b$12$nCyu4bU8OIZO8OIGm8tzMOR/zIB7L6bs1QSkN7.S425i5FTc4mQhm
11 $2b$12$IkWPJr2AxVLihCnQjQWCL.Tp9pSM421Oz/lcHMHNi57teU2jO2VDa
33 $2b$12$k0G8P5XDncdEyC2QfoI.xeQkJp3RgKaC5HzyFQnVNS0RdNraw.ht6
44 $2b$12$Uca8HoDxjjW65.zuNxAfpORRTfWNn6YhUajuE.cC/ya6Mhu6kngIa
mm $2b$12$dppVWkwAmeLqjJjY/mClkuxIrE7nwAb4xtStwDiJ23dtiUurlslnW
maaopooiuiuoi $2b$12$n1LcURQL1Ym9AzCeYY8HJuhj3AXiejGvKJqvcCPVrTtsF5V4jhfWW
999 $2b$12$rUtgBj.zaZ.QpTwIzt55S.KFs4niwo78ulZC1eZD/2xaAvvfzCwJG
997 $2b$12$KXPfK7YHcKPavPLEejXJl.4H05VMGcJYvQlI2Q4LL1UhxxvZdBNZW
996 $2b$12$76sGS4U08r6/2w9zPy1dputjME5uCkKRwZo4PkhTTldLlFWSnEtKO
995 $2b$12$bJlbjb6cp0VC0F1CrbBxNuu7EzNGzXKcbOjizKets4WZpHx7xh0eW
user100 $2b$12$uYIU72zuIk2XX5PkvecFYOQeRJLhKLZBOhoTZCNfkY6znFB2urfQy
user/ $2b$12$/hJEg9Tun/Da8K34PYL3tOJE2PNn4Cqjp2isb9hA50w6qKagb7rdu
user< $2b$12$v07kbuH3/3F4g0CmEPEbWeTpaPt2MFsemVNhyrY1tiaWwrPY1UaMm
user.. $2b$12$89EXK8SNraw.STKbvvp.ZerYJFUbOvHB3qzd1W/9sT6K15CwrTtLW
user56 $2b$12$J1mkH83gcCd7DSFDRI2xg.V2rfgE4l1ydTxvEvVApieAclkIODlq6
user101 $2b$12$p0xyUvmnHz15mzxa7KwSG.Yh/RbfFhkklOp6hxP9pKXNJyolHOBmG
user102 $2b$12$iRyiRFD3t/zUQHvwUqlh4OurwgJIk9zO0cZ8X902MBwvUVUC9mPPC
user34 $2b$12$eB72ytjj3cVzTNBasE5tLeBQQ5/68lf48ArFG3kMfuw5Ldhw6BJYW
user1 $2b$12$r2iCZbJLqt9ySuHFQtRB2OOzLY9NidSq186nZ8raWuaSNy5YAAkIW
user200 $2b$12$PPXfhO67M1dCSN7Osw6MVuy6V8As7FPRsdSnsLVM6TzMN0Qy8Mo6O
user1000 $2b$12$T0t6umOX9hOjYxKIsmlyNOSCc3sG79QOReP0iIYvjo6bqRnHzdQTq
user1001 $2b$12$BhLVPLhTq4eTYlFeW4pGWOZPd1W6w84JQKxKA3ZoNqyP01i2ZSBk.
```

All user passwords are hashed and stored. The usernames are on the left. Hashed passwords are on the right.

This file is stored in Google Cloud.

10. Chat logs

Our application has the ability to log chats into a file. The logged messages are listed in order that they were sent. The timestamps are provided and the encrypted messages are listed as well. For each message, you will see the sender and receiver.

```
[2025-05-03 19:15:38] 17 -> @user100 7dl8ZHp+qhglXJhAUwpoXw==
[2025-05-03 19:20:19] user100 -> @17 7dl8ZHp+qhglXJhAUwpoXw==
[2025-05-03 19:20:19] user100 -> @17 7dl8ZHp+qhglXJhAUwpoXw==
[2025-05-03 19:20:19] user100 -> @17 uDtYJ69keGlX2RlGlr7YOw==
[2025-05-03 19:20:19] user100 -> @17 7dl8ZHp+qhglXJhAUwpoXw==
[2025-05-03 19:20:43] 17 -> @user100 7dl8ZHp+qhglXJhAUwpoXw==
[2025-05-03 19:26:07] user100 -> @17 TFR2KuWb5pjIjRv9/s6WAg==
```

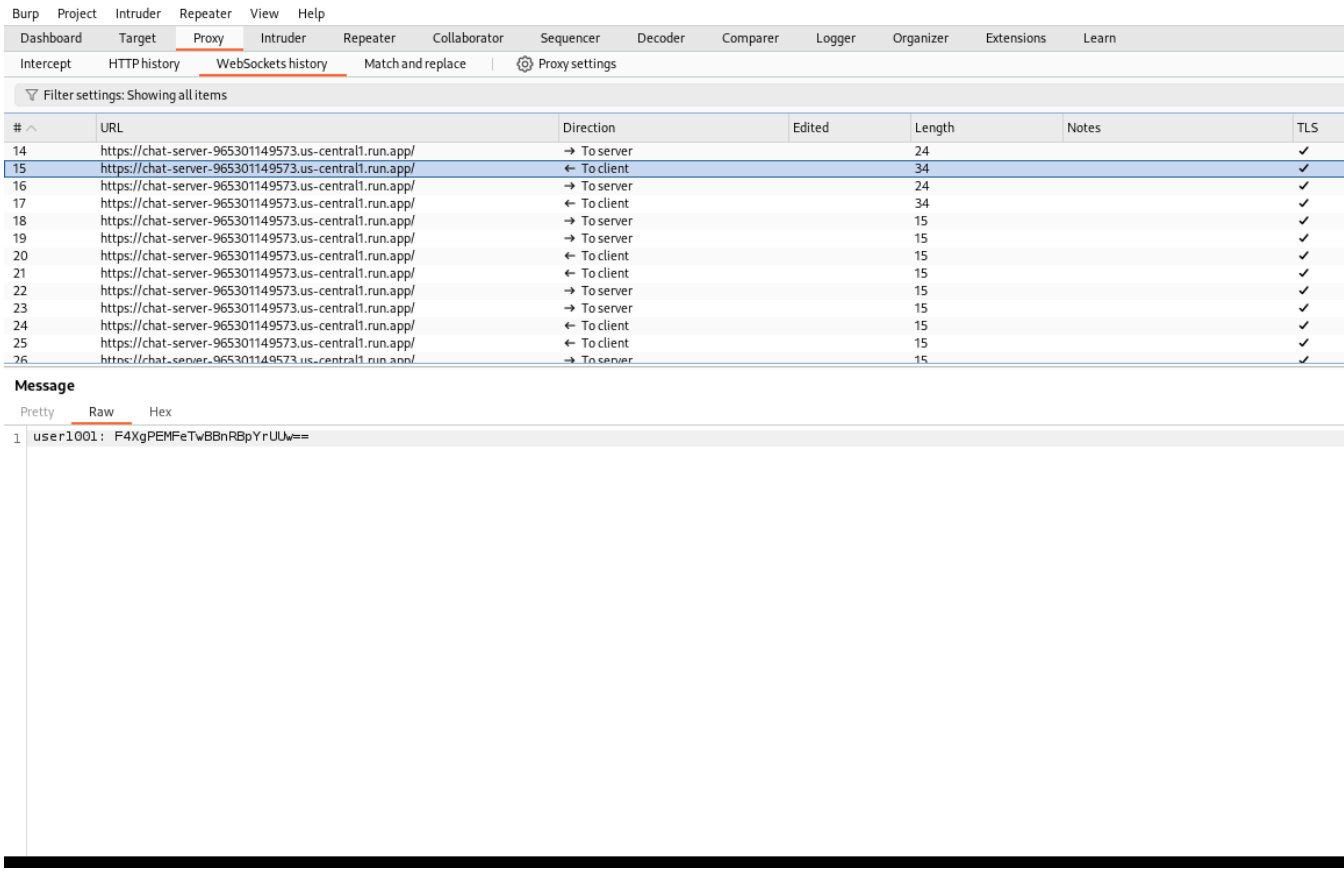
Chat log (from Google Cloud) shown between user100 and 17. The timestamps and the encrypted message (for security) are shown.

11. Encryption

Our end-to-end encryption uses AES and a key for encryption and decryption. The client encrypts the message with the key. When the message is on the server, it is still encrypted. Once it reaches the other client, the other client decrypts it to plaintext. Our key is kept secret so we made sure that it is not hard coded into the code.

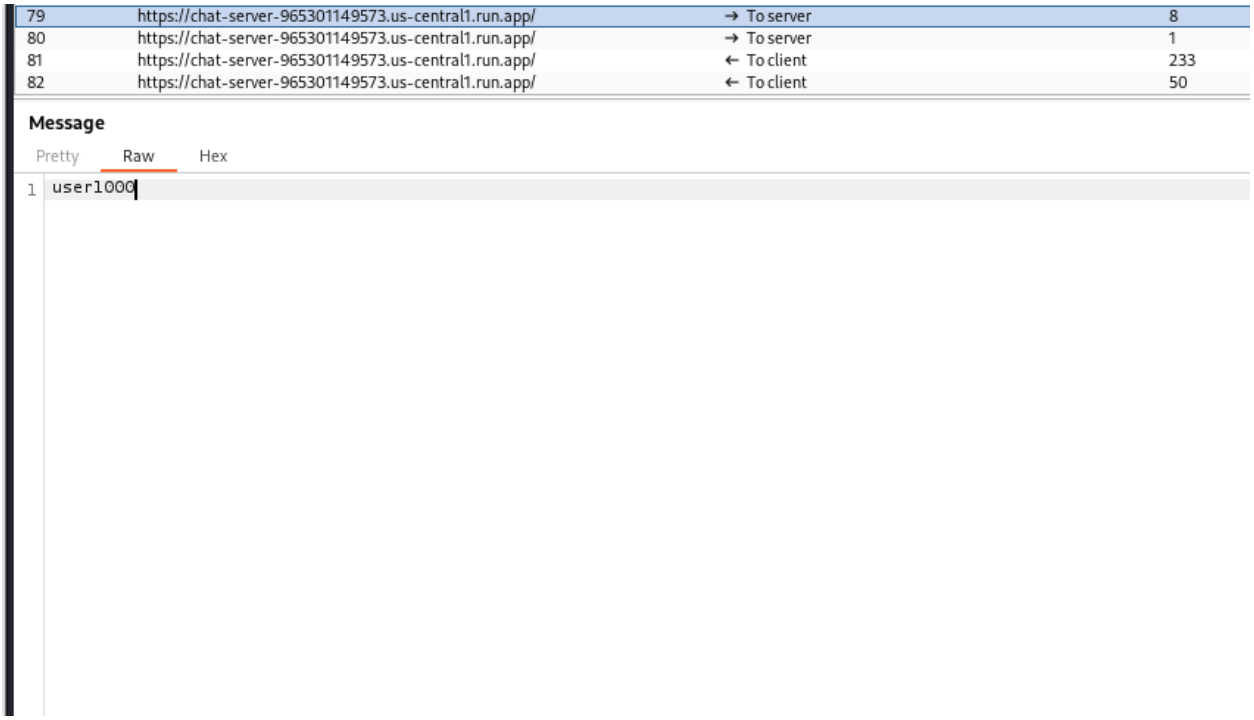
```
[2025-05-03 19:50:58] user1001 -> @user1000 7dl8ZHp+qhgLXJhAUwpoXw==
[2025-05-03 19:51:00] user1000 -> @user1001 pImVSloYavpIzCMfENMfzCzjJdnfdXwOW3yamIju40eMgJKocvDvggdbEZR2qFvyfLCNDMbtzZs+tC9XF63+JPwdoi+jLcJqtY
```

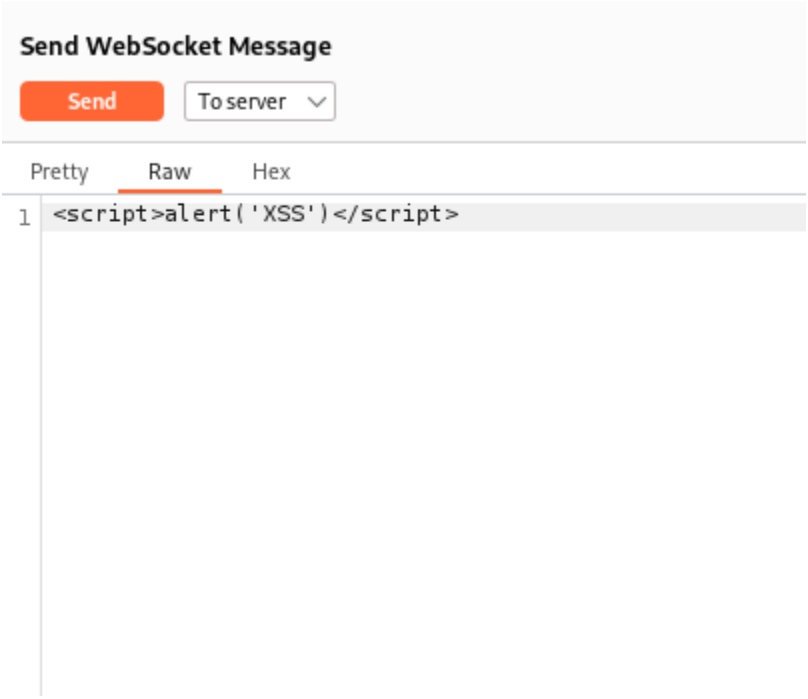
The logging functionality occurs in the server, so it shows that the messages in the server are encrypted. The long encrypted message is a file, which shows that files are encrypted as well.



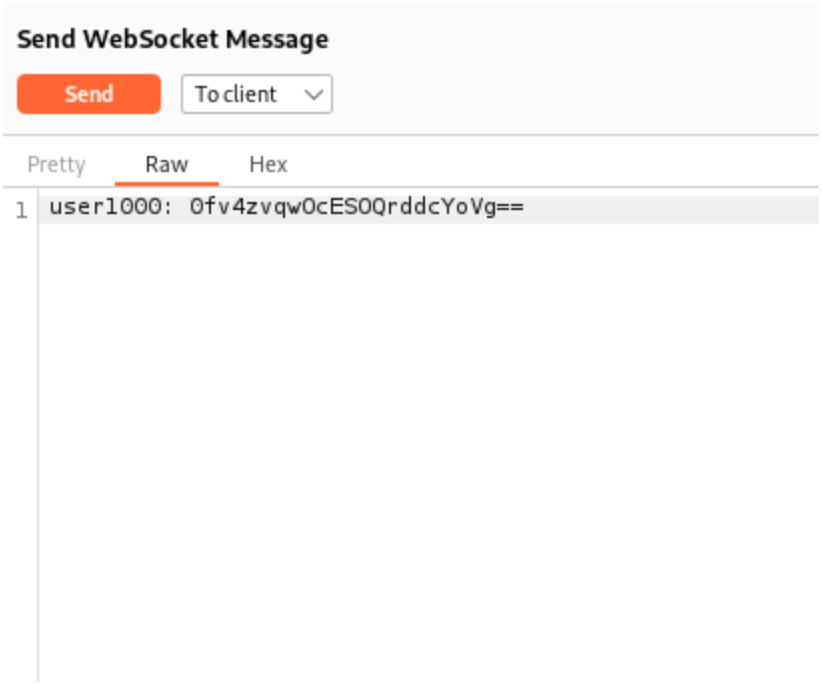
Burp Suite captures the message, and you can see that it is encrypted.

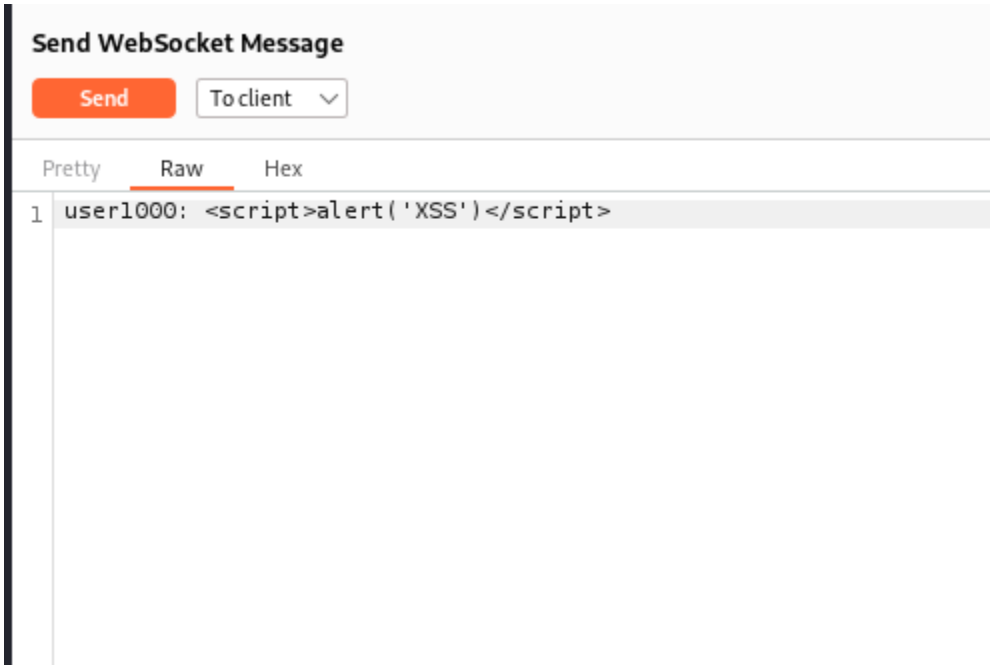
12. Input Sanitization

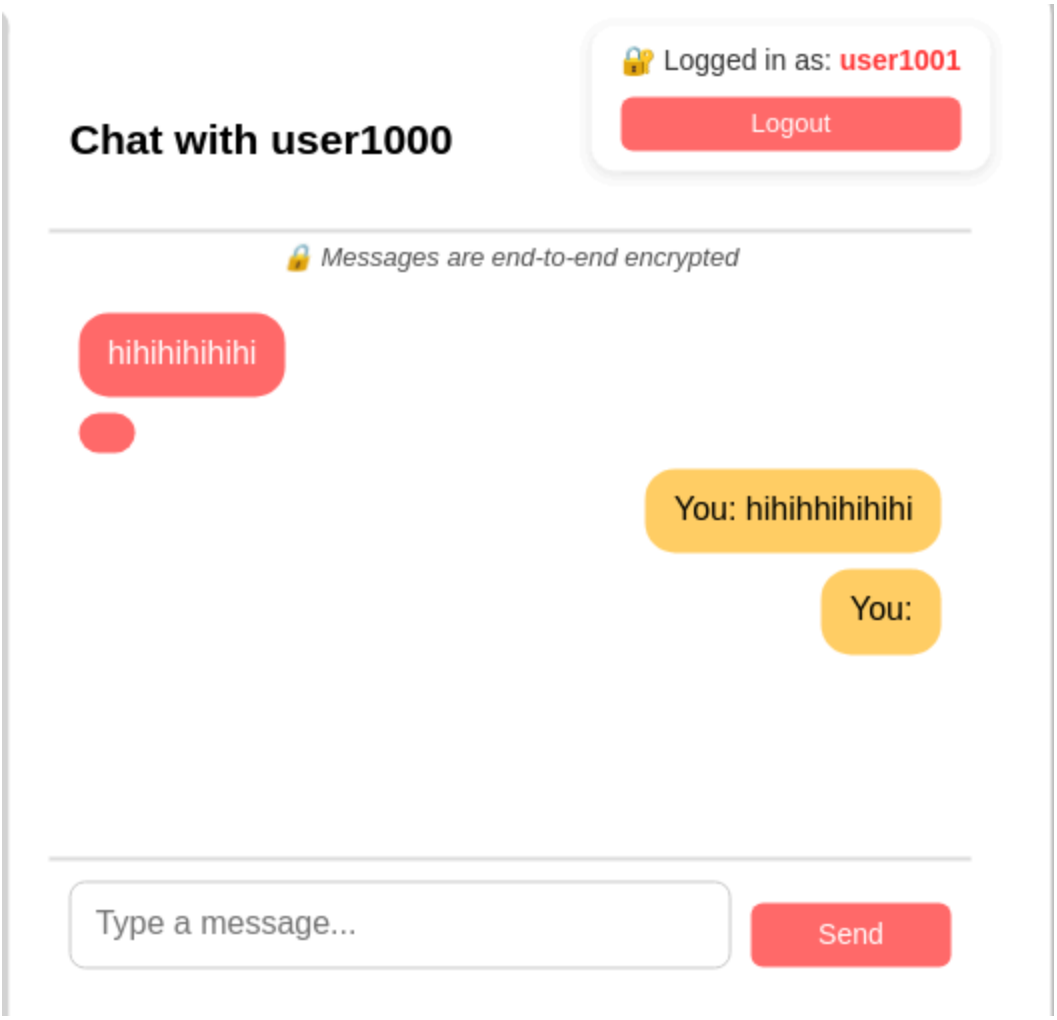




In Burp Suite, I modify the username input in the login page and send it. I found no issues.

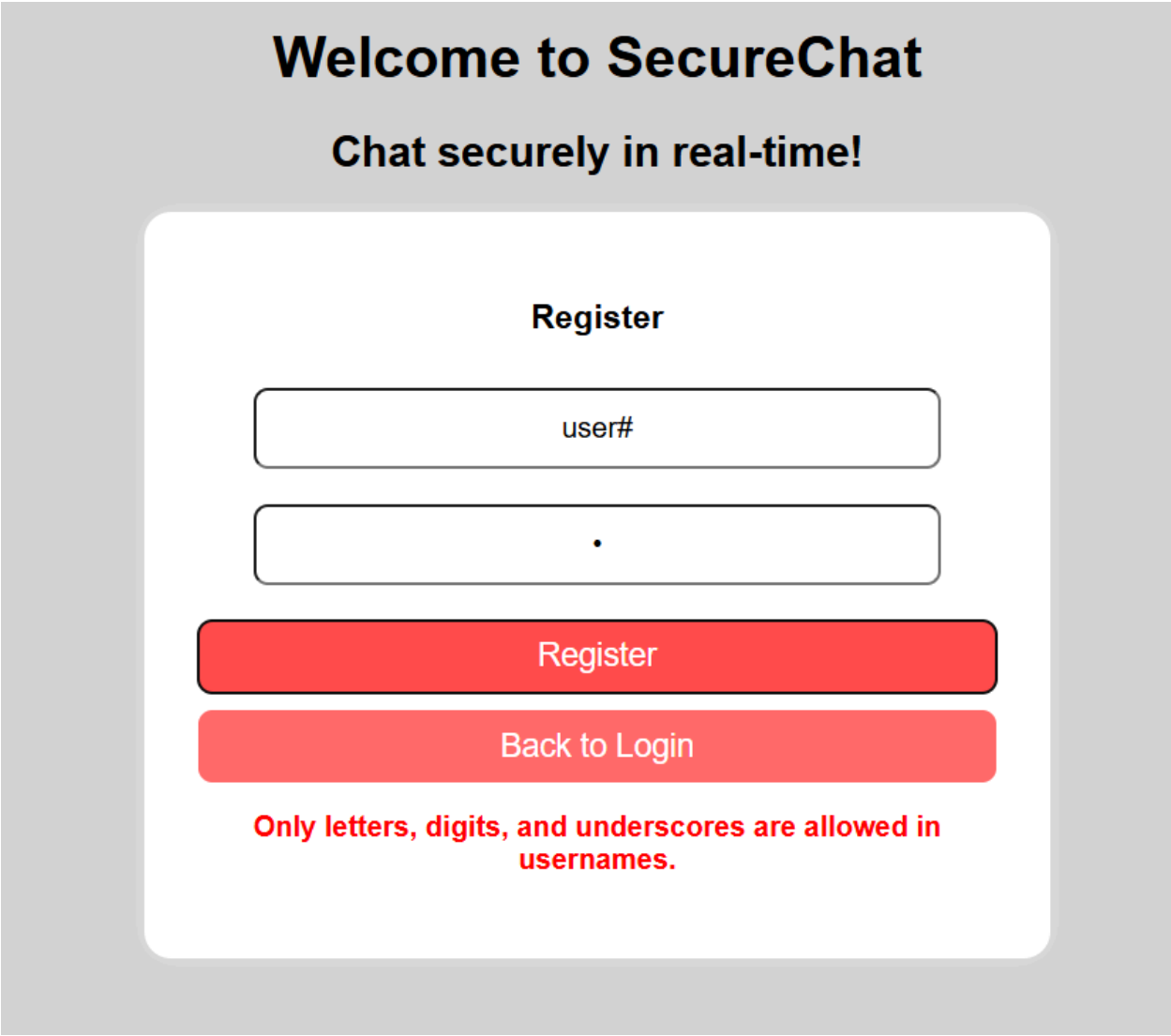

Send WebSocket Message
Send
To client
Pretty
Raw
Hex
1
user1000: 0fv4zvqwOcESOQrddcYoVg==


Send WebSocket Message
Send
To client
Pretty
Raw
Hex
1
user1000: <script>alert('XSS')</script>



In Burp Suite, I captured a sent message and modified it in the repeater and sent it. Due to sanitization, it doesn't harm our system.

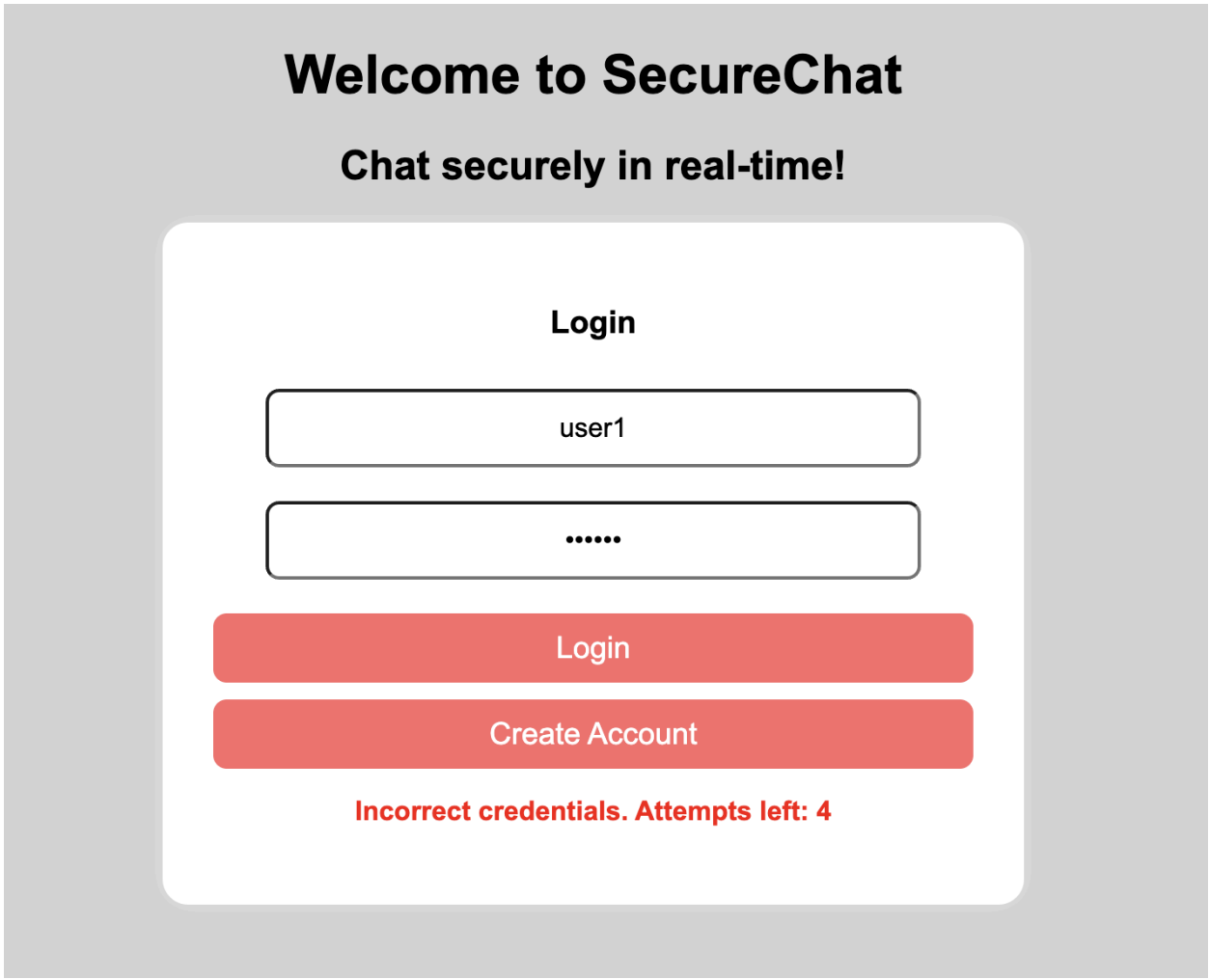
A sanitization policy we have is that usernames can only contain letters, numbers, and underscores.



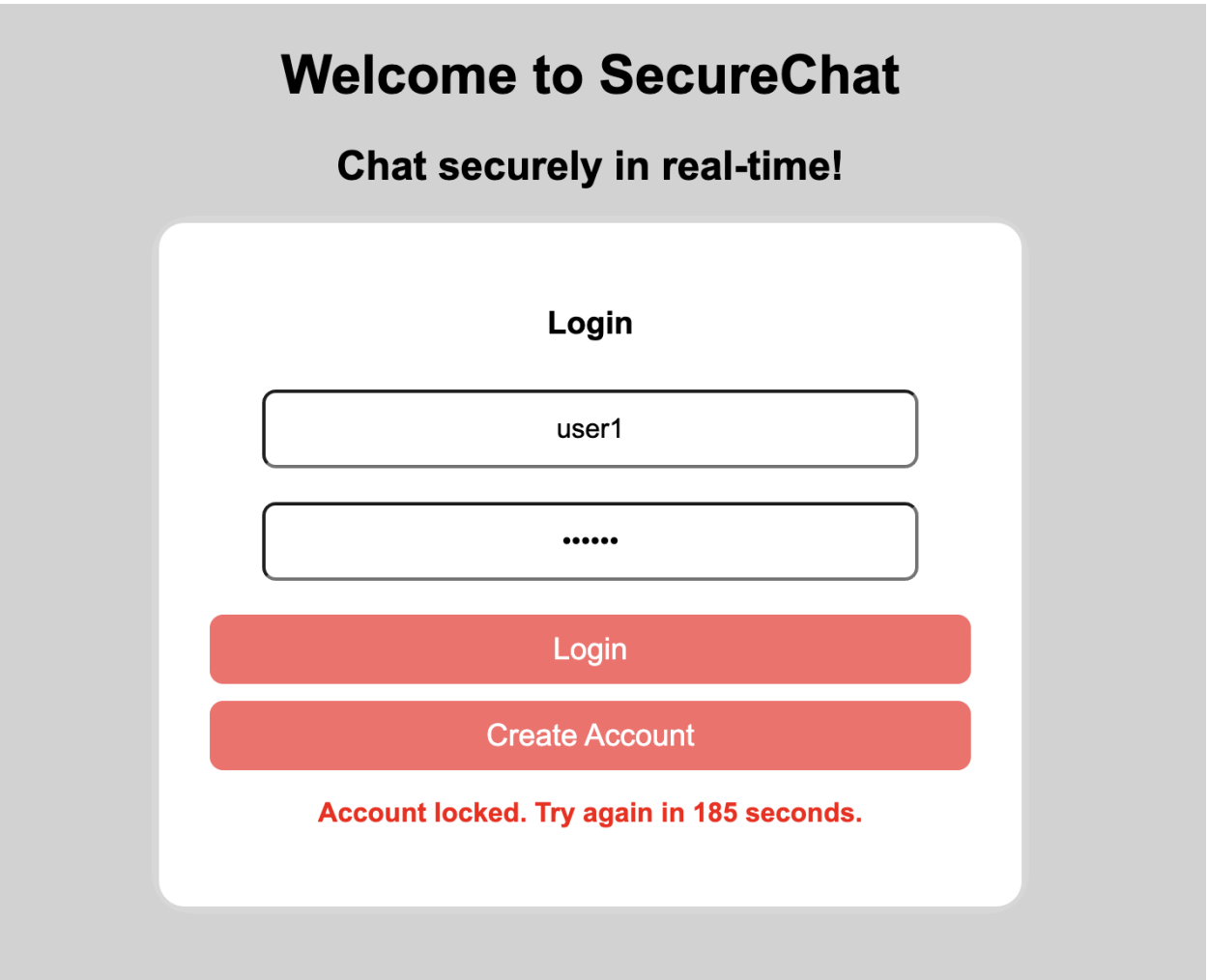
To prevent any kind of injection or XSS, this policy is being enforced. The username in the image cannot be registered or used in the login page.

13. Log-in/Registration attempts limitation.

We implemented a feature to prevent brute-force login attempts. If a user attempts multiple login attempts with the wrong password, a timer counts down after each failed attempt. After a certain number of failed attempts, the user will receive a timeout.

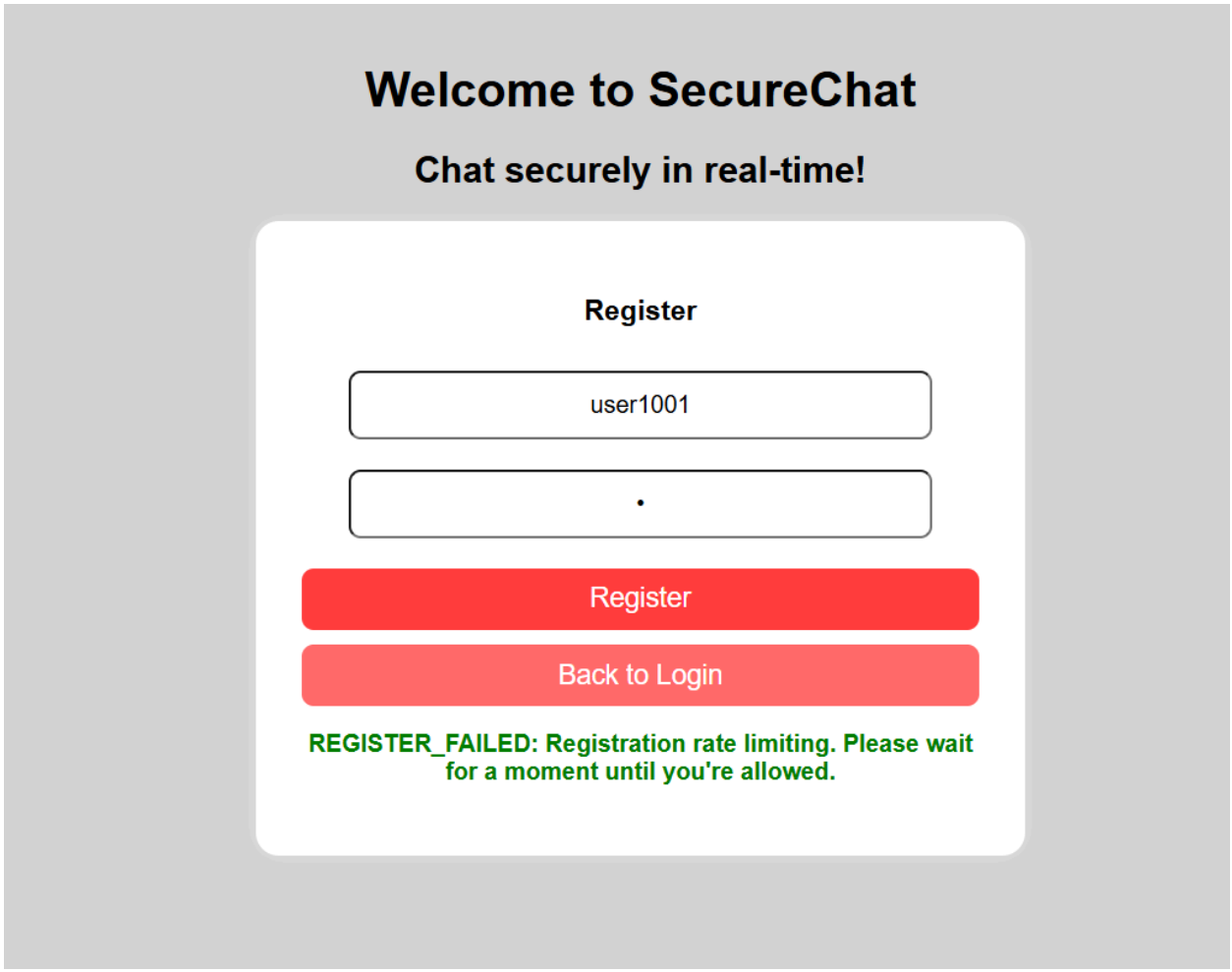


The user gets 5 attempts to log in.



After five failed attempts, the system will put a 5-minute timer on the user's account.

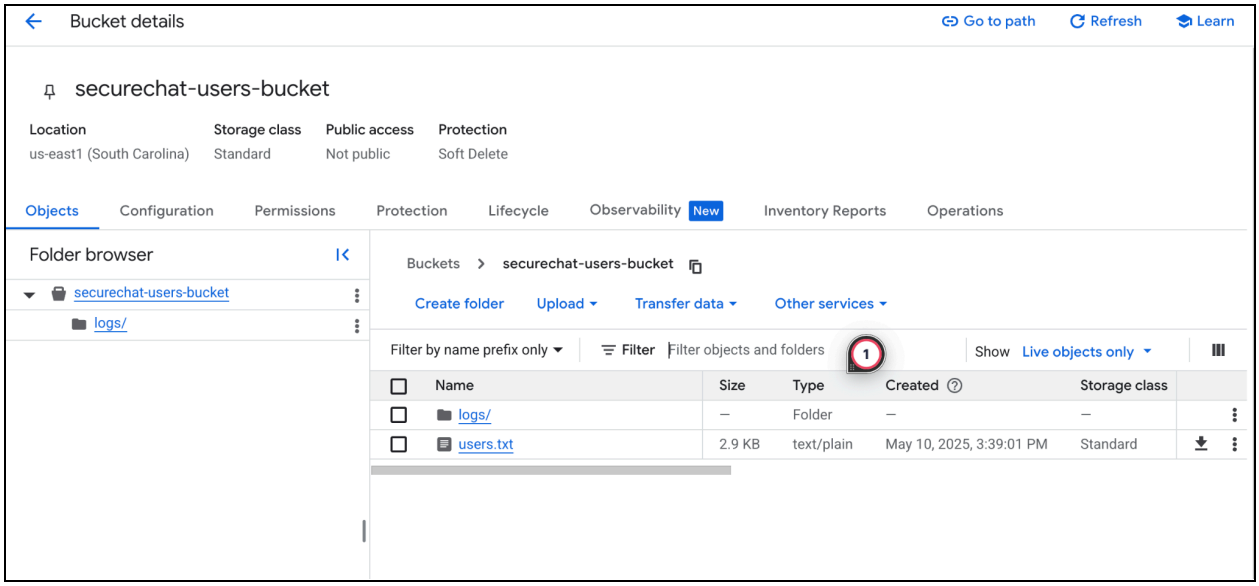
Our registration limit policy usually limits users to register only one account within a 15 second time period. This is meant to ban bot registration attacks.

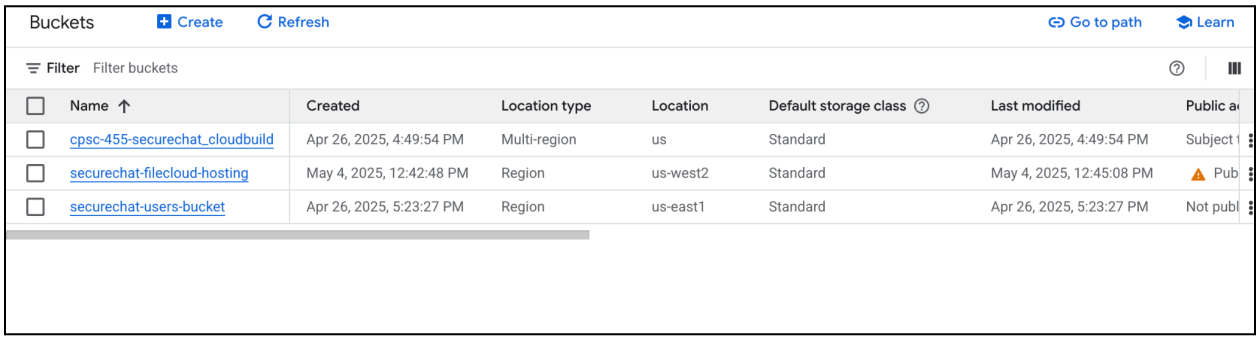


User1001 cannot be registered in the above example because of our registration rate limiting policy.

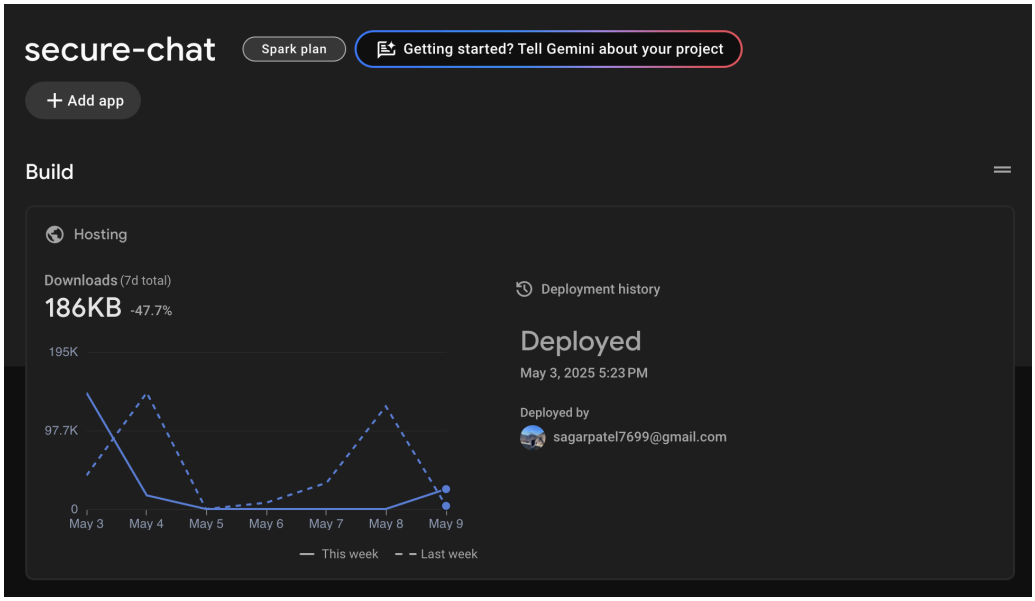
14. Cloud infrastructure for front and back end

We used Google Cloud for hosting our back-end, as well as storing our user.txt (text file for user credentials) and user log. Part of file sharing is lso hosted on Cloud.





Our front end is hosted on Firebase:



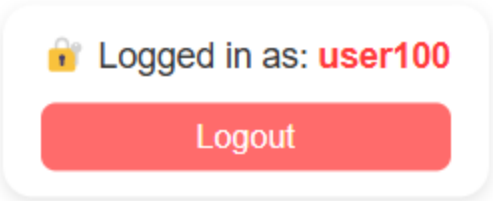
15. Extra Features

These are features that are not required to be implemented, but we decided to add it in for a better user experience.

Notification Feature

The user is notified that there is one new message from 17. Clicking on 17 makes the notification disappear.

Logout Feature



At the top right shows your username and a logout button you can use after you are tired chatting.