# Client-Server operations Application

This is a simple client-server chat application that allows users to communicate with each other over a network and perform command operations such as MSGGET, MSGSTORE, SHUDOWN, LOGIN, LOGOUT, QUIT.

## Methods Description

Here are some key methods used in the code:

Client.java

### getMessage (String serverResponse): This method is responsible for receiving a string serverResponse as its parameter and then printing it to the console. It serves as a utility function for displaying messages of the day received from a server.

### storeMessage (String serverInput, String userInput, DataInputStream dataIpStr, DataOutputStream dataOpStr, BufferedReader input): This method checks whether the user is logged in. If the user is logged in, it accepts their input as a new "Message of the Day" and updates it accordingly. If the user is not logged in, it prints an error message to notify them of their unauthenticated status.

### quit (DataInputStream dataIpStr, DataOutputStream dataOpStr, String serverInput, Socket clientSocket, boolean keepRunning): This method exits the application and closes any open socket for client when confirmed by server.

### shutdown (DataInputStream dataIpStr, DataOutputStream dataOpStr, String serverInput, Socket clientSocket): This method closes all sockets and exits the application at both client and server's end.

Server.java

### get\_message (DataOutputStream os, String [] motd, int active\_motd, int motd\_count, Socket clientSocket): This method is responsible for printing the upcoming message of the day to the user and reading the next message for access.

### storeMessage(DataInputStream is, DataOutputStream os, String[] motd, int motd\_count, String line, String active\_user, Socket clientSocket): This method checks the user's authentication status. If the user is logged in (authenticated), it captures their input as a new Message of the Day and updates it. In the case of unauthenticated users, it prints an error message, ensuring that only authenticated users can modify the message directory file.

### login (DataOutputStream os, String active\_user, String line, Socket clientSocket): This method logs in the user by verifying their credentials against the records in the usersDir.txt file and issues an error message if the user is already logged in.

### logout (DataOutputStream os, String active\_user, Socket clientSocket): This method manages the logout process for logged-in users and handles error messaging for those not logged in.

### quit (DataOutputStream os, Socket clientSocket): This method is responsible for displaying an acknowledgment message to the user just before they close the socket connection.

### shutdown (DataOutputStream os, boolean keep\_going, String active\_user, Socket clientSocket): This method, when invoked by a user with root privileges, shuts down the server by closing sockets and streams, effectively ending the application.

### chatHandler (): This method manages communication with an individual client, encompassing the exchange of messages, both sending and receiving, to facilitate seamless interaction.

## File Descriptions

Here's a brief description of the key files in this project:

### Server.java - This file contains the server-side code for our application. It handles incoming client connections and message routing.

Usage: Modify this file to customize server functionality.

### Client.java - This file contains the client-side code for our application. It allows a single user to connect to the server and exchange messages with other clients.

Usage: Customize this file to add features or functionality to the client.

### messageDir.txt - This directory is utilized to store and manage messages that can be updated by authenticated users

### usersDir.txt - This file contains a directory that stores user profiles and related information, facilitating the management and access of user-specific data within the system.

## Contributions

Here's a list of individual contributions to this project:

1. Arati Khanolkar - https://github.com/yourusername

- Implemented the client-side functionality.

- Fixed the bug related to message handling.

- Wrote the initial version of the README file.

- Optimized the code for better performance.

2. Soumya Vaidya - https://github.com/contributorusername

- Developed the server-side functionality.

- Added unit tests for message validation.

- Improved error handling and logging.

- Updated the documentation for the README file.

## Known Bugs

### Bug 1: Commands accepted only in capital letters

- Description: While user puts commands on client terminal only CAPITAL case commands are accepted

- Status: Unresolved

- Workaround: User should provide commands only in uppercase format.

### Bug 2: Newline on client terminal for commands - MSGGET with 200 status

- Description: We are getting an extra newline on the client terminal because writeUTF is adding the newline.

- Status: Unresolved

## How to Compile

### Create a directory for the programs.

### When creating a directory, make sure it has the following files:

Client.java

Server.java

Makefile

### Open a window terminal and navigate to folder where files are store with the command:

“cd Private”, “cd lastname\_firstnameinitial\_p1”

### Run the makefile by typing in the command, “make Server.class Client.class ”,

### Compile both Server and Client java files using commands, “javac Server.java”, “javac Client.java”

## How to Run

### Open new terminal and run the Server by using command“java Server”

### In the second terminal window now run the Client by typing command, “java Client”

### It will display the connection established notification on both terminals

### User can now enter commands to execute MSGGET, MSGSTORE, SHUDOWN, LOGIN, LOGOUT, QUIT   
 operations

## Sample Output

### Client Terminal

[soumyasv@login:4 ~/Private/cis527p1/server-client]$ java Client

Connected to Server.

MSGGET

200 OK

Try to be a rainbow in someone else's cloud.

MSGGET

200 OK

You do not find the happy life.

MSGGET

200 OK

Sometimes you will never know the value of a moment, until it becomes a memory.

MSGGET

200 OK

Happiness depends upon ourselves.

LOGIN root root01  
200 OK

LOGIN john john01

You are already logged in as root

LOGOUT

200 OK

LOGIN abc abc1

410 Wrong UserID or password

LOGOUT

405 You are not logged in.

SHUTDOWN

402 User not allowed to execute this command.

MSGSTORE

200 OK

Please input new Message of the day to be added.

HELLO WORLD

429 Limit exceed for storing message: maxLimit=20 messages

LOGIN john john01

200 OK

QUIT

200 OK

Thank you for using the server.

### Server Terminal

[soumyasv@login:1 ~/Private/cis527p1/server-client]$ java Server

client port54948

141.215.69.202 connected

Logged in successfull, User: root

user root logged out.

Login failed for the user

Logged in successfull, User: john

User disconnected.