Low Level Design

Instant Chatters

Group - 1
Vibhu Bhardwaj
Licency Patel
Sukanya Pradhan
Laxmi Kant Singh
Jangam Jonas Edward

Revision Number: 3.0
Last date of revision: 18-10-2022

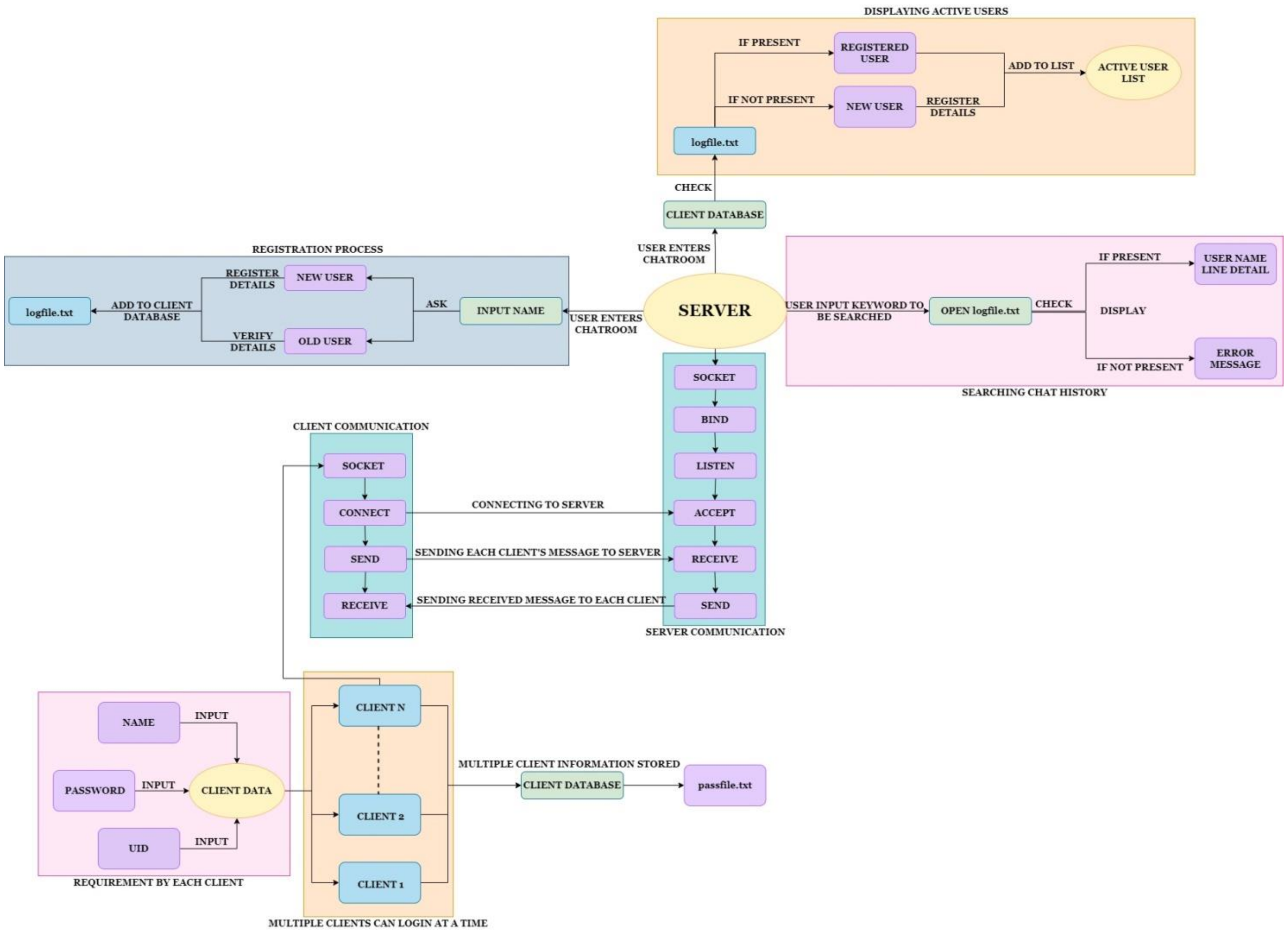
Contents

1. Introduction

1.1. Goals

- The role of a server is to store, send and receive data. It provides services to the clients that get connected to it.
- It has a structure sockaddr_in that stores the information about the server and a client_t structure that stores the client's information.
- As usual, the server binds to a particular IP address and port no. and waits for connections from clients.
- It accepts the request from the client that they want to get connected with and all of them get added up in a queue.
- The server has a record of all the clients that are connected to it. If the client is a new user, the server sends a message for registration.
- It provides two services to the user. A user can either have a private chat with another client or a large group depending on with whom he shares the Ip and port.
- Here, Multithreading is used to handle multiple clients with the help of POSIX threads.
- We declare a variable called tid, which is of type pthread_t, which is an integer used to identify the thread in the system. After declaring tid, we call the pthread_create() function to create a thread. The pthread_join() function for threads is the equivalent of wait() for processes.
- A mechanism known as "thread synchronization" prevents two or more concurrent processes or threads from running a specific program segment known as a "critical section" at the same time. When the lock is set, no other thread can access the locked region of the code and release it when done with the execution.

2. Dataflow diagram



3. Modules

3.1 SERVER SIDE MODULES:

3.1.1 Removing the next line character from the messages.

Name	str_trim_lf()		
Input	**Parameters-** Array, length	**Type-** Character pointer, Integer	**Comments-**The function takes an array and its length as an input
Output	Stores the string without the ‘\n’ character.	Void	Replaces ‘\n’ character with ‘\0’
Process	This function is to trim the next line character from the buffer.		

3.1.2 Clear Output Buffer

Name	str_overwrite_stdout()		
Input	Parameters -	Type-	Comments- No input
Output		void	-
Process	Its purpose is to clear (or flush) the output buffer and move the buffered data to console or disk.		

3.1.3 Adding up all the clients joined in a queue.

Name	queue_add()		
Input	Parameters- Pointer of type client_t structure	Type- Client_t	Comments- This function takes a pointer of type client_t structure as an input.
Output	-	void	Updates all the clients according to the client_t structure.
Process	The function Add clients to clients array		

3.1.4 Removing the clients from the existing queue.

Name	queue_remove()		
Input	Parameters- Userid(uid)	Type- integer	Comments-. This function takes the user id as input.
Output	None	void	Comments-
Process	This function removes clients from the clients array.		

3.1.5 Send messages to the clients

Name	send_message()		
Input	Parameters- s(pointer) uid status	Type- Character Integer character	
Output	-	void	-Message gets sent to other clients having same status
Process	This function is used to send messages between clients who all have the same status except the sender.		

3.1.6 Handling client communication

Name	handle_client()		
Input	Parameters- arg(client's structure which is a structure pointer)	Type- void	
Output	None	-	-
Process	This function handles all client communication (sending and receiving messages and files) and also helps store chat history into a file.		

3.1.6.1 Validating client

Name	handle_client()		
Input	Parameters- arg(client's structure which is a structure pointer)	Type- void	
Output	None	-	-
Process	This function will ask the user whether the user is registered or not. If user is registered, open passfile.txt and search for the matching user name and password. If the match is found, user can enter the chatroom or else login denied.		

3.2 CLIENT SIDE MODULES-

3.2.1 Clear Output Buffer

Name	str_overwrite_stdout()		
Input	Parameters -	Type-	Comments- No input
Output		void	-
Process	Its purpose is to clear (or flush) the output buffer and move the buffered data to console or disk.		

3.2.2 Removing the next line character from the messages.

Name	str_trim_lf()		
Input	**Parameters-** Array, length	**Type-** Character pointer, Integer	**Comments-**The function takes an array and its length as an input
Output	Stores the string without the '\n' character.	Void	Replaces '\n' character with '\0'
Process	This function is to trim the next line character from the buffer.		

3.2.3 Receiving signal to interrupt

Name	catch_ctrl_and_exit()		
Input	Parameters- -	Type- -	-
Output	-	void	-
Process	The function enables the client to log out from the session.		

3.2.4 Sending messages to the server

Name	send_msg_handler()		
Input	Parameters- None	Type- int	Comments-
Output	None	-	Makes use of the socket to send messages from the client to server after connection has been made
Process	This function is used to send messages to the server from the client.		

3.2.5 Receive a message sent from the server

Name	receive_msg_handler()		
Input	Parameters- None	Type-	Comments
Output	No output by itself	void	Makes use of the socket to receive messages from the server after connection has been made
Process	This function enables the client to receive the message from the server.		

3.3 SEARCH CHAT HISTORY

Name	main()		
Input	Parameters- None	Type-	Comments
Output	None	-	Makes use of the logfile.txt to search chat history when a user enter a keyword.
Process	This function will search the chat history file for a particular keyword entered by the user and if found, is displayed.		