

Teacher Student Interface

A COURSE PROJECT REPORT

By

Vanshika Singh Gour (RA2011033010096)

Vama Pachori (RA2011033010115)

Jaspreet Singh (RA2011033010092)

Saurabh Kumar (RA2011033010084)

Under the guidance of

Dr. T R Saravanan

In partial fulfilment for the Course

of

18CSC302J - COMPUTER NETWORKS

in Department of Computational Intelligence



FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

Kattankulathur, Chengalpattu District

NOVEMBER 2022

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Under Section 3 of UGC Act, 1956)

BONAFIDE CERTIFICATE

Certified that this mini project report " **TSI (Teacher Student Interface)** " is the bonafide work of **Jaspreet Singh (RA2011033010092), Vama Pachori (RA2011033010115), Vanshika Singh Gour (RA2011033010096), Saurabh Kumar (RA2011033010084)** who carried out the project work under my supervision.

SIGNATURE

Dr T R Saravanan
Assistant Professor,
Department of Computational Intelligence
SRM Institute of Science & Technology, Kattankulathur

ACKNOWLEDGEMENT

We express our heartfelt thanks to our honorable **Vice Chancellor Dr. C. MUTHAMIZHCHELVAN**, for being the beacon in all our endeavors.

We would like to express our warmth of gratitude to our **Registrar Dr. S. Ponnusamy**, for his encouragement.

We express our profound gratitude to our **Dean (College of Engineering and Technology) Dr. T. V.Gopal**, for bringing out novelty in all executions.

We would like to express our heartfelt thanks to Chairperson, School of Computing **Dr. Revathi Venkataraman**, for imparting confidence to complete my course project

We wish to express my sincere thanks to **Course Audit Professor Dr. Annapurani Panaiyappan, Professor and Head, Department of Networking and Communications** and **Course Coordinators** for their constant encouragement and support.

We are highly thankful to our Course project Faculty **Dr. T R Saravanan, Assistant Professor, Department of Computational Intelligence**, for his assistance, timely suggestion and guidance throughout the duration of this course project.

We extend my gratitude to our **HoD Dr. R Annie Uthra, Professor and Head, Department of Computational Intelligence** and my Departmental colleagues for their Support.

Finally, we thank our parents and friends near and dear ones who directly and indirectly contributed to the successful completion of our project. Above all, I thank the almighty for showering his blessings on me to complete my Course project.

INDEX

Sr. No.	TITLE	Page No.
	Abstract	5
1	Introduction	6
2	Objective	6
3	Requirement Analysis	6
4	Architecture and Design	7
5	Module Description	11
6	Algorithms	12
7	Functions and Constructors	15
8	Input and Output	17
9	Project Preview and Screenshots	17
10	Conclusion	20
11	References	20

Abstract

Practical sessions always play a very important role in School, University Etc. If the we consider fields such as Engineering, then the practical must be taken very carefully, because practical knowledge always leads to better understanding of that subject and if there is not better understanding between speaker and listener i.e., student and teacher then student may get some difficulties while doing given practical work.

In Today's world we can observe that students are tending towards social network learning / E – Learning and digital medium for studying So the concept of TSI **Teacher Student Interface** is digital (computer-based communication) medium basically created for practical related to computer i.e., computer based practical. TSI will help to improvise the communication between student and teacher, the interface will be computer so it will become a lot easier for teachers to give attention to every needy student within time. If this kind of interaction is generated then overall workload to learn the concepts in practical will gets reduced.

TSI can be created through local network by using SOCKET PROGRAMMING. It can be used in several computers to create connection between them. TSI also includes different features such as admin passkey, Add-Remove client, Send-Receive Files (program code). It is going to be very suitable solution for teaching method in practical.

1. Introduction

This software can be used to make simple Interface between students and teacher so that the interaction can be done between them while performing live activities such as programming practical, problem solving sessions. In Schools / Colleges, Computer based practical always requires timely guidance by respective practical teachers. Teachers must make sure that every student is understanding the session. But due to time restrictions, student personal problems, etc. some students do not understand that session.

2. Objective

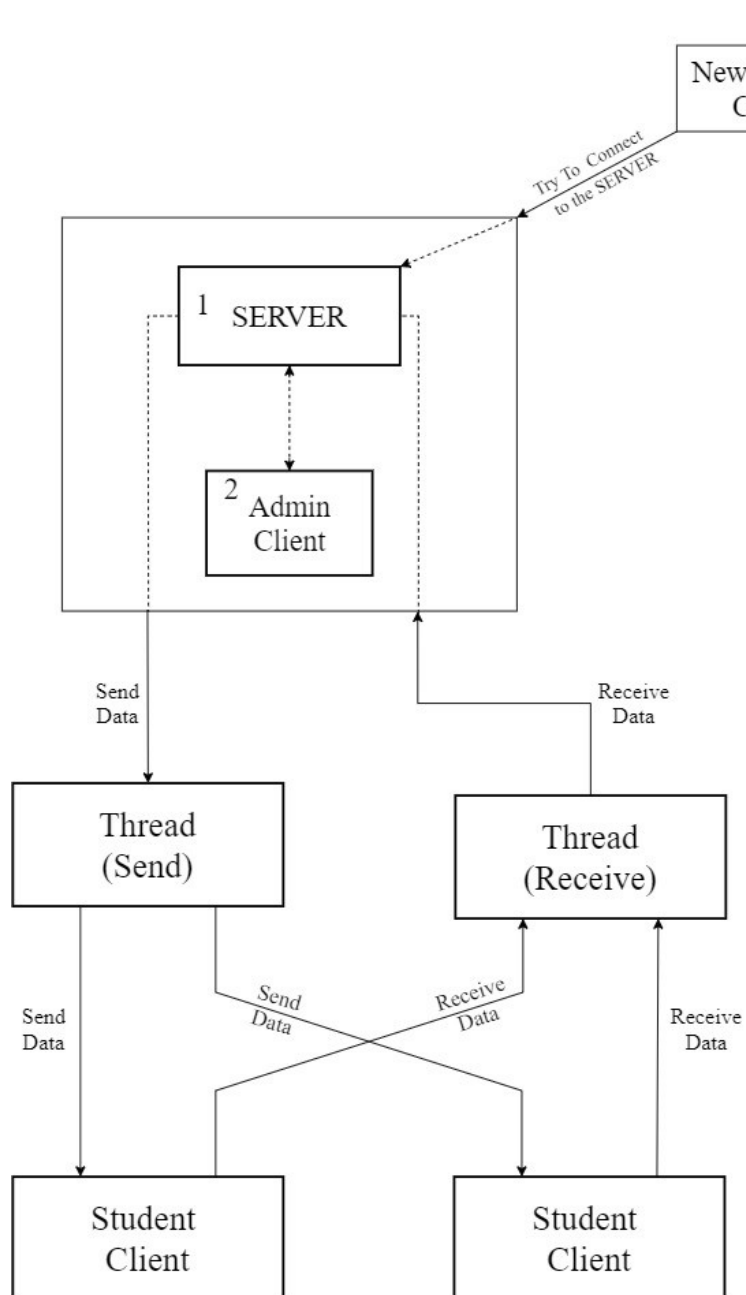
- **INTERFACE** - Create two ways interface between Students (client) and Teacher. (server)
- **DISCUSSION** - Discussion is established between students and Teacher. (Through Chats)
- **QUERIES** - To solve problems in easy manner by understanding queries from students.
- **SHARE** - Sharing of file resources to faculty. (server)

3. Requirement Analysis

- **Hardware:** Desktop machines/laptop, LAN connection.
- **Software:** Any Version of Ubuntu (LINUX) with C / C++ compiler.

4. Architecture and Design

4.1 Control Flow Diagram



Working of main system program:

The server and admin client must be running on same machine. And obviously the student clients can be run on different machines connected in LAN.

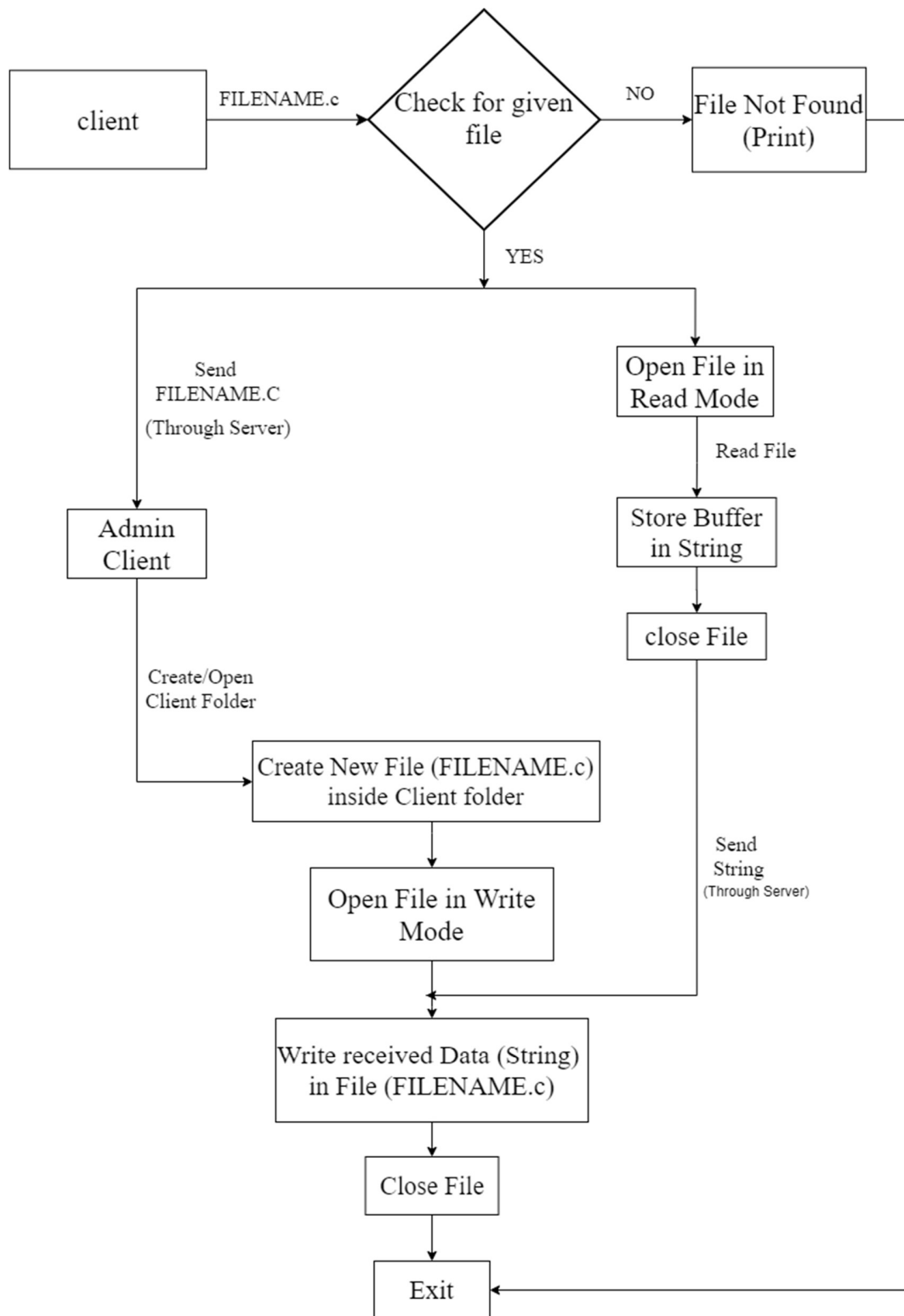
Initially the server starts with its IP address (refer process 1). Then the admin client will connect to this server (refer process 2). After that the other student clients can connect to the server (refer process 3).

After that student clients and admin client can communicate with each other. When a student sends a message/request to server, it is processed at the server and according to the request the message is forwarded to the other student clients and admin client.

After activating the server, it initiates two threads (processes), one for receiving the message from either admin client or student (or at a time both) and other to send the message to other clients (either admin client or student clients).

The message received by server from a client is processed at server site, if it is a requested command then, its output is given to that client (or broadcasted to other clients is necessary), and if it's a normal text message then it is processed accordingly and passed to the recipients (other clients or admin).

4.2 Flowchart for Process of sending file:



At first, the client will give file name and then at client side, it will check if the file is present or not. If the file is not present then, failure will be shown.

If it is present, then it will open file in read mode. then, it will read the contents of tile, and it will store it in a string. it will send the name of the file to another client or server

After sending the file name, server will create a file with the same name which is received by the client. Then server will open the file in write mode and then, it will receive the string from the client in which, the client has stored its file content.

The server will use this string to write into the file which has created recently. After writing into the file, it will store the client.

The process is similar while sending file from Server to Client.

5. Module Description

ADMIN LOGIN:

When teacher connects to the TSI's main server, it can declare itself as an admin to the server by giving a particular passkey. After which no one can become admin anymore, as this TSI's server only supports one admin at a time

ADD/REMOVE CLIENT:

Students are clients, they can request to get connected or disconnect whenever they want. There is a queue in our server which can store the details of multiple clients (such as Name, Client ID, IP address details)

SEND/RECEIVE DATA:

Student(clients) can send either message or his/her program file to teacher(admin).

6. Algorithms

6.1 Add and Remove Clients

6.1.1 For Adding

Step 1: Accept the connection from client using newsockfd, then check for Maximum number of clients reached or not? If not then continue else go to step 5.

Step 2: Declare client pointer for client.t structure.

ASSIGN

cli -> address = clientaddr,

cli -> sockfd = newsockfd,

cli -> uid = uid++.

Step 3: REPEAT Step 4 for i = 0 to MAX_CLIENTS.

Step 4: IF clients [i] is false.

Then assign clients [i] = cl.

Step 5: print message maximum clients connected and no more clients can join and exit.

6.1.2 For removing

Step 1: REPEAT Step 2 for i = 0 to MAX_CLIENTS.

Step 2: IF clients [i] is true

Then If clients [i] == uid

Then clients [i] = NULL.

6.2 Admin Login

Step 1: IF client sends 'set – admin' message to server

Then server will ask client to Enter Password.

Step 2: The client will enter the password and send it to the server.

Step 3: Server then checks and compares the passwords received from that client.

Step 4: IF Password is correct

Then set client as Admin and server sends message to that client as ‘You are now admin’

To other clients it will send message as ‘The particular client has become admin’.

ELSE

The sever will send message ‘Wrong password’ to that client.

6.3 File Sharing

6.3.1 From server (Admin client) to client

Step 1:IF admin client sends message 'send' to server.

Then server asks that admin client for the name or Roll number of students.

Step 2:The admin client enters and send the name or roll number to server.

Step 3: Server then checks for if any clients with name or roll number has joined and currently active or not If there is one,

Then server asks admin client for file name to be sent.

Step 4: Admin client sends file name to the server.

Step 5: Server checks if File is present or not

IF present

Then

server opens the file and reads all content then stores it in a buffer array. After that it sends this buffer array to the target client (whose Roll number or name was given by Admin client).

Else

Server will ask for name again from admin client by sending message “File can't be opened, please enter file name again”.

Step 6: When targeted client receives the filename and buffer array. It creates a file with same filename and opens it then it writes the file using the buffer array.

Step 7: Exit.

6 .3.2 From client to server (Admin client)

Step 1: If client sends message "send" to server.

Then server asks that client for filename.

Step 2: Client enters filename and then checks if the file with given name present or not.

If file is not present then

Client is asked to enter the name of file again.

Else

File name is sent to the server and the contents of the file are read and stored in buffer array and is sent to the server.

Step 3: Server creates "Server Files" directory and inside that it creates the directory named after the current active admin-client's name (if current admin-client is not present then the directory is named as "Admin" by default) inside that directory it creates another directory named after client's Name or Roll Number inside which the server creates a file with the same filename which has been received from client and server then writes into this file using contents of received buffer.

Step 4: Admin client sends file name to the server.

Step 5: Exit.

7. Functions and Constructors

- **send_f () and recv_f ():**

These are the functions available at client side, which are responsible for transmission of data(msg, file) as per the conditions they contain , each condition leads to some action like to receive the file or to send the file or to simply send the string as the message on the server to broadcast it.

- **main ():**

At client side, this function is used to get the client info like is client a student or teacher, then its name and roll no.

This function creates socket and establishes socket connection, then these data is shared through socket connection with the server. And then this main function uses 2 threads to run 2 functions in background at a time those are send_f() and recv_f(). On detection of the flag value equals to 1 this function will close the connection and end the program.

At server side, this function create socket and creates connection with the client. Whenever new clients want to join this main function checks for the number of connected clients are full or not and accordingly accepts the connection. After accepting the connection from client it uses a thread to run a function for recently connected client which is handle_client().

- **void handle_client ():**

In server side this is the very important function as it handles each and every client and its action.

The data shared between clients is firstly comes to the server and how to handle it is done with this function.

This function has certain conditions for incoming messages from clients, if they meet certain conditions that part of instructions is executed and accordingly the client gets its output.

- **void queue_add (client_t *cl):**

This function is called by main function in order to add the recently connected client to the list of connected clients.

- **void queue_remove (int uid):**

This function is used to remove the client that has been disconnected from the server from the list of connected clients.

- **void send_message (char *s, int uid):**

This function is used to send the received message from a client to every other client except the one who send it to the server. In short, this function is responsible for broadcasting a message on the server.

- **char *nameOfAdmin ():**

This function is used to get and return the name of currently active ADMIN on the server. If there isn't one then this function returns string "Admin".

- **void sendFileToNonAdmin (char *fileName, int i):**

This function is responsible for transfer of File from ADMIN CLIENT(SERVER) to CLIENT. It uses some file handling functions to achieve that.

- **int checkStudent (char *s, int uid):**

This function is used to check the given name or roll number of students is present inside the server's list of connected clients or not. If present then it returns its number from the list but if not present then it returns value 0.

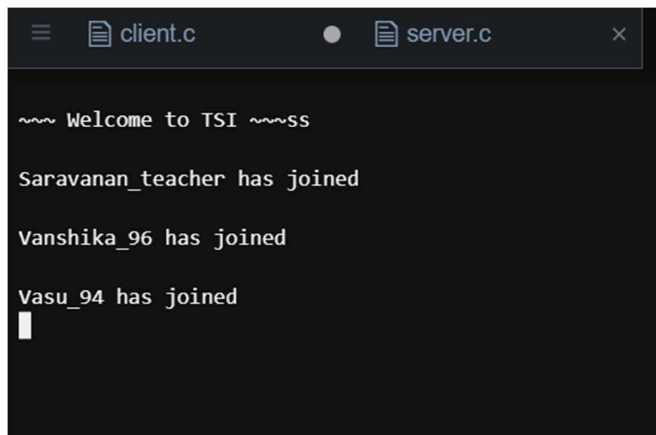
8. Input and Output

Input: While executing server or client we have to give port number and IP address after connection sent from any clients will be taken as input in the server for further processing.

Output: According to received input string Server checks the received input string (message) from client, if it meets certain conditions, it may proceed further to give output as a broadcasted message on server to all the clients or transmission of file (either client to server or server to client).

9. Project Preview and Screenshots

9.1 Clients Joining



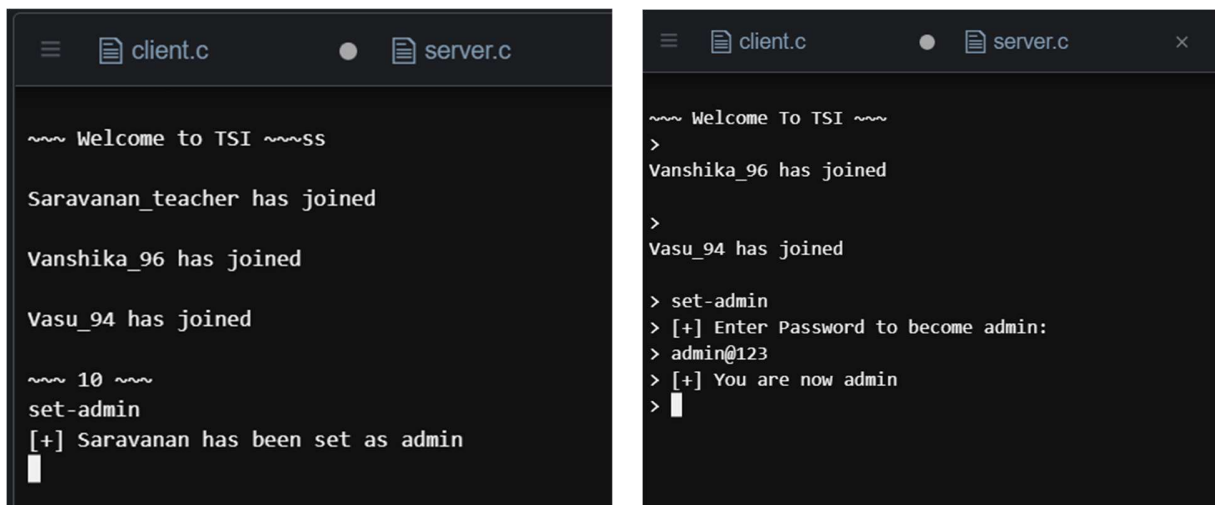
```

client.c  server.c

~~~ Welcome to TSI ~~~ss
Saravanan_teacher has joined
Vanshika_96 has joined
Vasu_94 has joined

```

9.2 Admin Login



```

client.c  server.c

~~~ Welcome to TSI ~~~ss
Saravanan_teacher has joined
Vanshika_96 has joined
Vasu_94 has joined

~~~ 10 ~~~
set-admin
[+] Saravanan has been set as admin

> set-admin
> [+] Enter Password to become admin:
> admin@123
> [+] You are now admin
>

```

9.3 File/ Data sharing:

9.3.1 Communication through Chat :

```

client.c  server.c

~~~ Welcome To TSI ~~~
>
Vanshika_96 has joined

>
Vasu_94 has joined

> set-admin
> [+] Enter Password to become admin:
> admin@123
> [+] You are now admin
> hlo everyone hru all
> Vanshika : hlo sir we r gud
> Vasu : hlo sir gud sir hru
> 

```

```

client.c  server.c  ./.ser

Saravanan_teacher has joined
Vanshika_96 has joined
Vasu_94 has joined

~~~ 10 ~~~
set-admin
[+] Saravanan has been set as admin

~~~ 10 ~~~
Saravanan : hlo everyone hru all

~~~ 11 ~~~
Vanshika : hlo sir we r gud

~~~ 12 ~~~
Vasu : hlo sir gud sir hru

```

```

client.c  server.c

~~~ Welcome To TSI ~~~
>
Vasu_94 has joined

> [+] Saravanan has been set as admin
> Saravanan : hlo everyone hru all
> hlo sir we r gud
> Vasu : hlo sir gud sir hru
> 

```

```

client.c  server.c

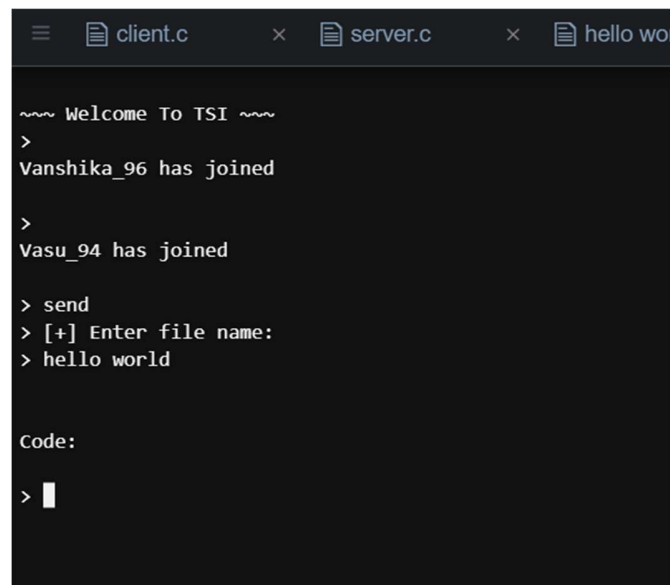
~~~ Welcome To TSI ~~~
> [+] Saravanan has been set as admin
> Saravanan : hlo everyone hru all
> Vanshika : hlo sir we r gud
> hlo sir gud sir hru
> 

```

9.3.2 File Sharing:

File sending from Client to server :

Client side



```

client.c  server.c  hello wor

~~~ Welcome To TSI ~~~
>
Vanshika_96 has joined

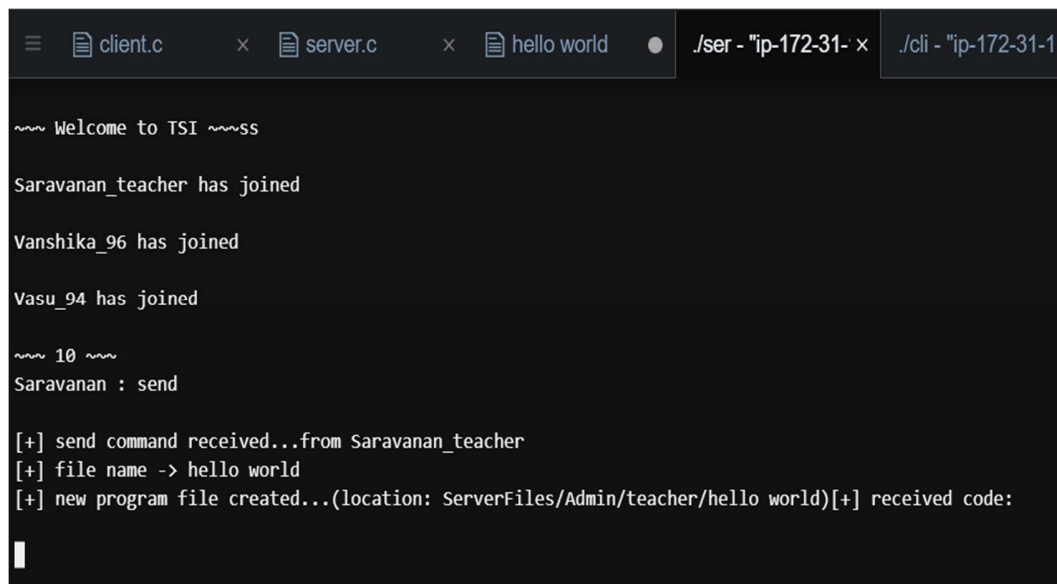
>
Vasu_94 has joined

> send
> [+] Enter file name:
> hello world

Code:
> █

```

Server side



```

client.c  server.c  hello world  ./.ser - "ip-172-31-1"  ./.cli - "ip-172-31-1"

~~~ Welcome to TSI ~~~ss
Saravanan_teacher has joined
Vanshika_96 has joined
Vasu_94 has joined

~~~ 10 ~~~
Saravanan : send

[+] send command received...from Saravanan_teacher
[+] file name -> hello world
[+] new program file created...(location: ServerFiles/Admin/teacher/hello world)[+] received code:
█

```

10. Conclusion

By using TSI, Interaction between student and client can be effectively carried out, means the students can chat with the teacher for guidance or can discuss any query related to his program/practical. Also, any student can share his program file through this software program with the teacher for checking. This way the interactive environment can be created among students and the teacher.

11. References

1. SOCKETS IN C - <https://www.geeksforgeeks.org/socket-programming-cc/>
2. Multithreading Concept - <https://www.geeksforgeeks.org/multithreading-c-2/>
3. Signals in C - https://www.tutorialspoint.com/c_standard_library/c_function_signal.htm
4. Command line Concept - <https://www.javatpoint.com/command-line-arguments-in-c>