**D.Y.PATIL COLLEGE OF ENGINEERING & TECH., KOLHAPUR**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**(2019-2020)**

****

**A**

**Mini Project**

**Report on**

**“TSI- TEACHER STUDENT INTERFACE”**

**Submitted by:**

**Roll No. Name of Student**

**38 Tejas Deepak Pandit**

**39 Jay Avinash Powar**

**51 Ajay Avinash Powar**

**Under the guidance of**

**Mr.A.S.Yadav**

**Class: S.Y.(CSE) Div: A Batch: S3**

**D.Y.PATIL COLLEGE OF ENGINEERING & TECH.,KOLHAPUR**

**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**Academic year 2019-2020**

****

**CERTIFICATE**

This is to certify that the mini project group consisting the following members have satisfactorily completed the Mini Project work entitled **“TSI – TEACHER STUDENT INTERFACE”** at SY BTech. (CSE) semester IV as prescribed in the syllabus of Shivaji University for the academic year 2019-2020.

**Name Exam seat no.**

1. **Tejas Deepak Pandit**
2. **Jay Avinash Powar**
3. **Ajay Avinash Powar**

**Date: 31/05/2020**

**Place:** Kolhapur

**Mr. A. S. Yadav** **Prof. B. D. Jitkar**

**(Project Guide) (HOD)**

**Prof. S. M. Surve Prof. Dr. K. V. Kulhalli**

**(Project coordinator) (Principal)**

**ACKNOWLEDGEMENT**

This Mini project work entitled “**TSI – TEACHER STUDENT INTERFACE**” was a formidable task, but with collective effort of our group and active guidance made it possible for us to complete.

First of all we would like to thank Prof. B.D.Jitkar (H.O.D., Department of computer science and engineering) delineating us with this Mini project work.

We would also like to thank Mini project coordinator **Name** for their support and interest that he has shown in bringing out this project and their guidance and cooperation We would also like to express our most humble and deepest gratitude to our Miniproject Guide **Mr.A.S.Yadav**, for providing us with the right guidance at the time of need it was for his presence and active guidance that we were able to complete the project work.

We would like to thank all our friends for their help, ideas, criticisms and also their encouragements for preparation of this project work. Any further ideas and constructive criticisms on our work shall be highly welcomed.

**Date: 31/05/2020**

**Place:** Kolhapur

**Name Sign**

1. Tejas D. Pandit
2. Jay A. Powar
3. Ajay A. Powar

**INDEX**

|  |  |  |
| --- | --- | --- |
| **Sr. No.** | **TITLE** | **Page No.** |
| 1 | Introduction | 5 |
| 2 | Problem Statement | 6 |
| 3 | Objective | 6 |
| 4 | Control Flow Diagram | 7 |
| 5 | Algorithm / Pseudo Code | 11 |
| 6 | System Requirements | 13 |
| 7 | Functions and Constructs | 14 |
| 8 | Input and Output | 15 |
| 9 | Conclusion | 20 |
| 10 | Reference | 20 |

**1) Introduction:**

Practical sessions always play a very important role in School, University, Etc. If the we consider fields such as Engineering then the practicals 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 practicals related to computer i.e. computer based practicals. 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 practicals 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 practicals.

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 practicals, problem solving sessions. In Schools / Colleges, Computer based practicals 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.

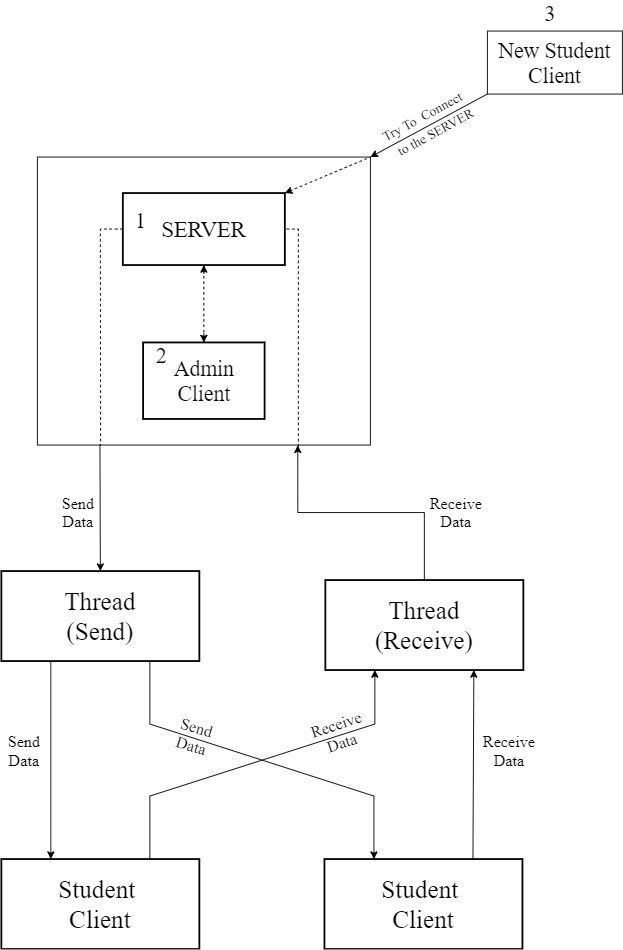
**Problem Statement:**

Implementing the socket programming (TCP) techniques to reduce the difficulties generating while practical sessions (related to computer practical's only).

**Objectives:**

* + **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)

**4) 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 ran on different machines connected in LAN.

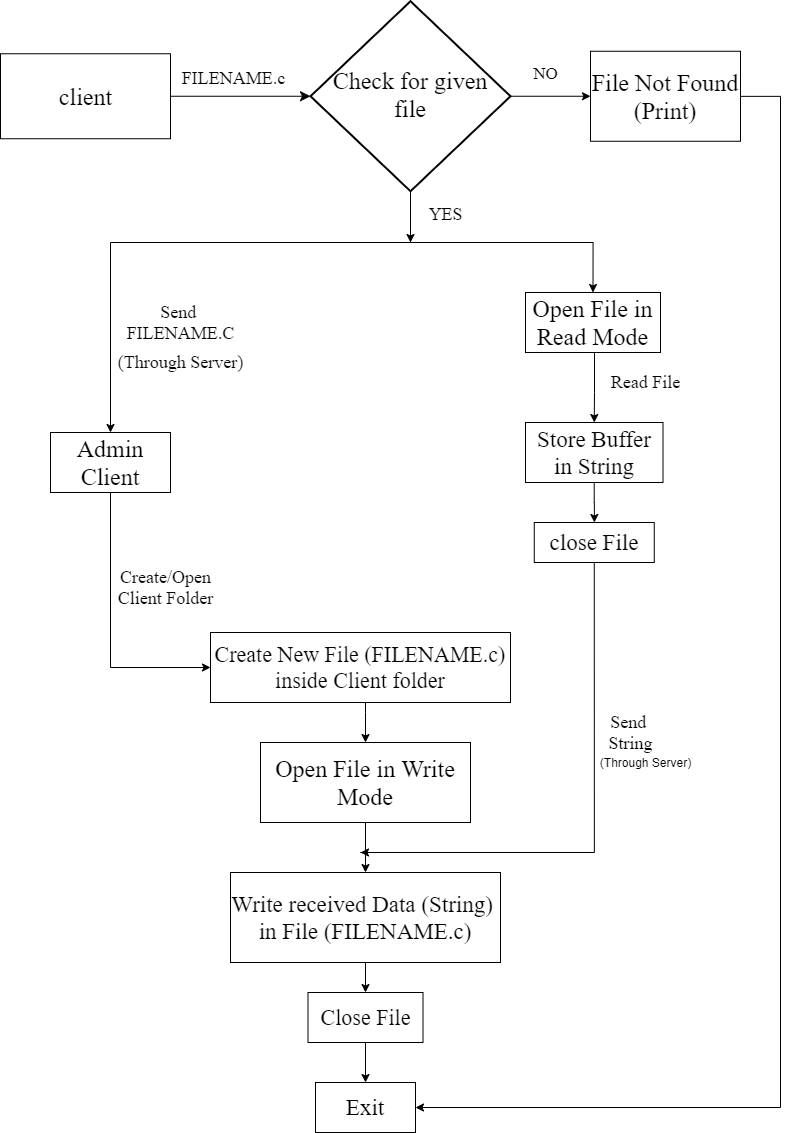
Initially the server starts with its ip address (refer process 1). Then the admin client will connect to this server (refer process 2). After that the other student clients can connect to the server(refer process 3).

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 its a normal text message then it is processed accordingly and passed to the recipients (other clients or admin).

* 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) Algorithms**

**A) Add and Remove Clients**

**a) 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.

**b) 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.

**B) 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 became admin’ .

ELSE

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

**C) File Sharing**

**a) 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 student.

**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.

**b) 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 "ServerFiles" 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 4:** Exit.

**6) System Requirements**

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

**7) Functions and Constructors**

**A) 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.

**B) 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 creat socket and creates connection with the client. Whenever new clients wants 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().

**C) 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.

**D) 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.

**E) 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.

**F) void send\_message (char \*s, int uid):**

This function is used to send the received message from a client to every other client execpt the one who send it to the server. Inshort this function is resposible for broadcasting a message on the server.

**G) 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".

**H) void sendFileToNonAdmin (char \*fileName, int i):**

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

**I) int checkStudent (char \*s, int uid):**

This function is used to check the given name or roll number of student 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

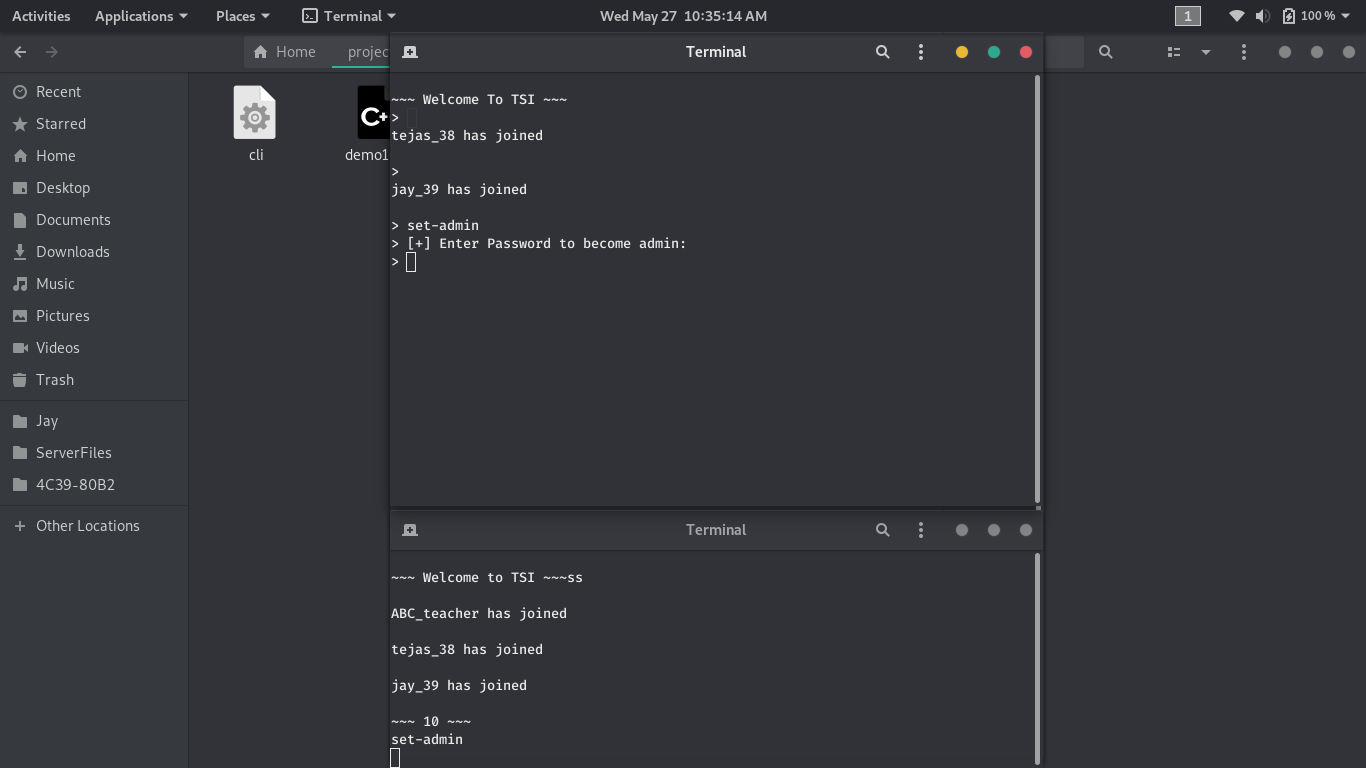
client , if it meets certain conditions it may proceed further to give output as a broadcasted

message on server to all the clients or transmission of file ( either client to server or server to

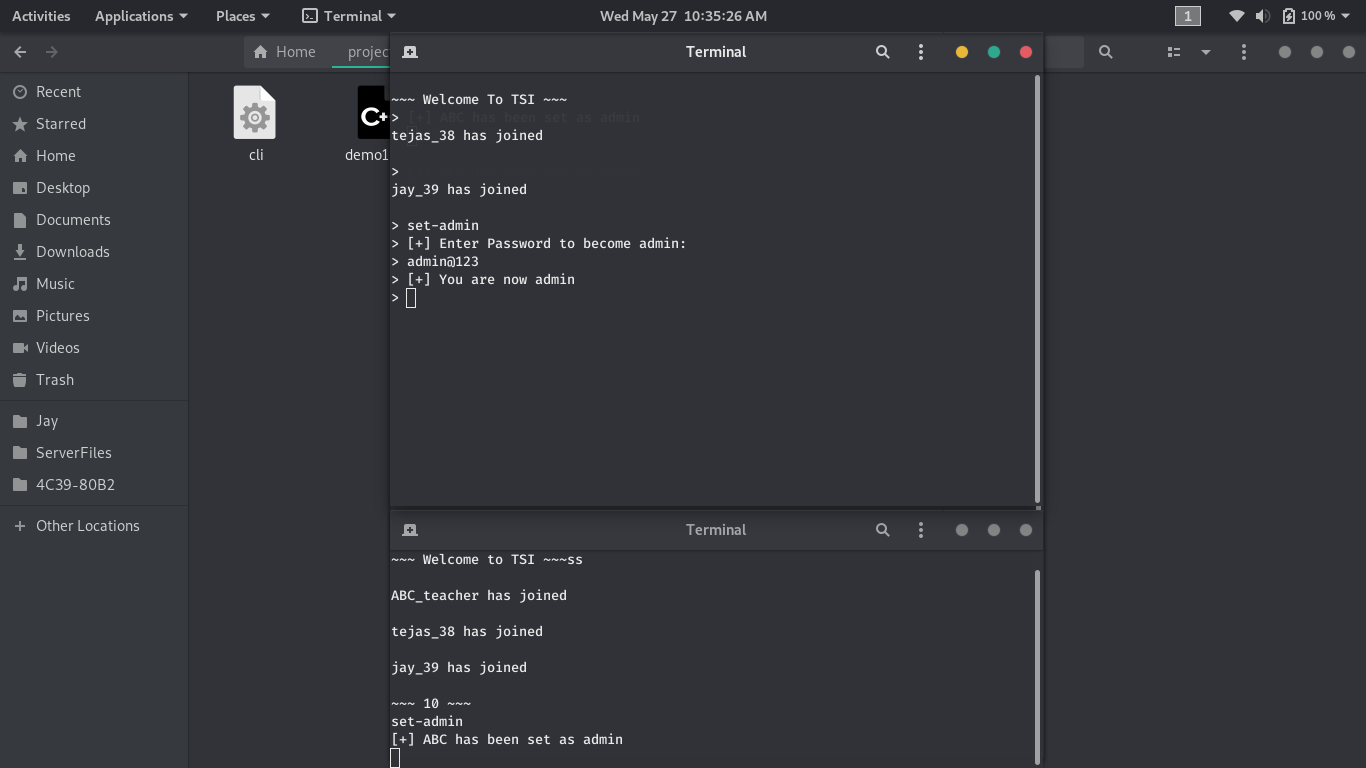
client ).

**EXAMPLES (Screenshots)**

**A) CLIENTS JOINING**

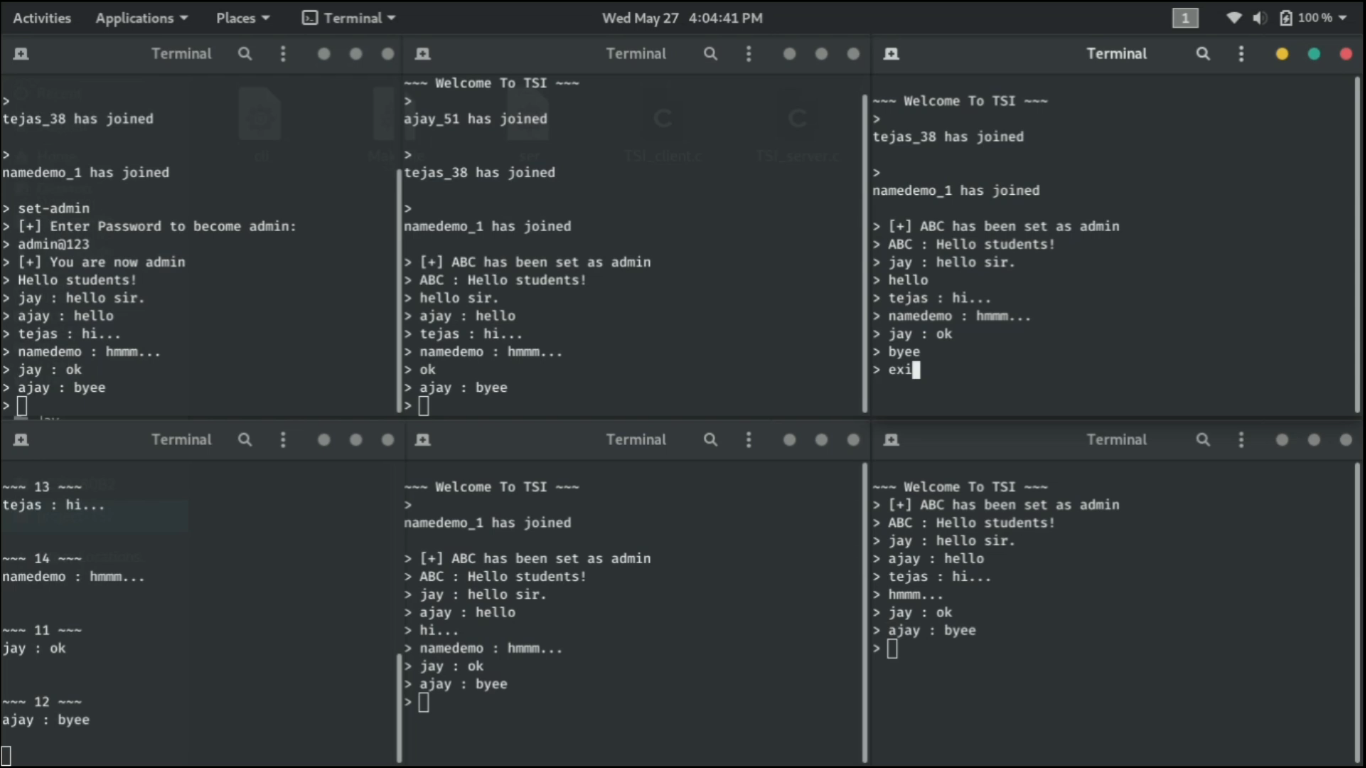


**B) ADMIN LOGIN**

****

**C) File/data sharing**

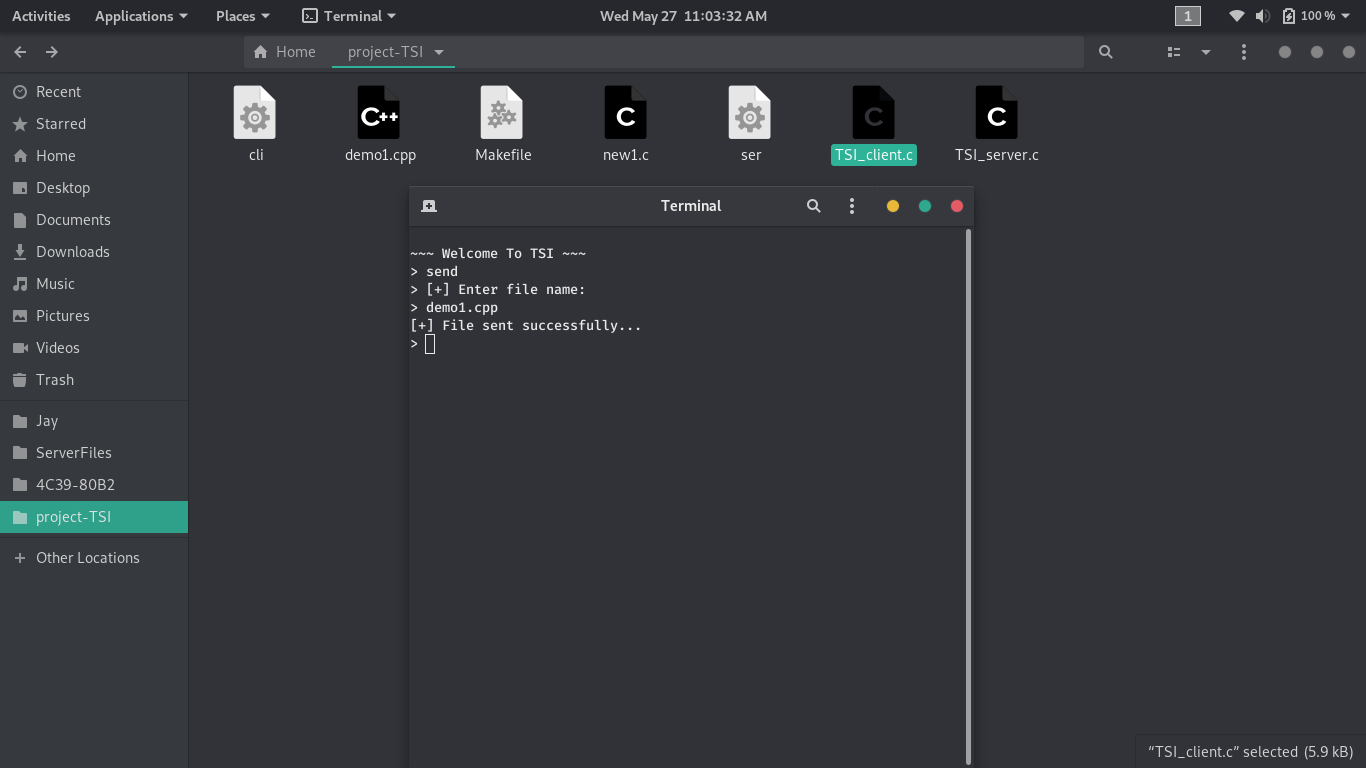
**1. Communication through Chat :**

****

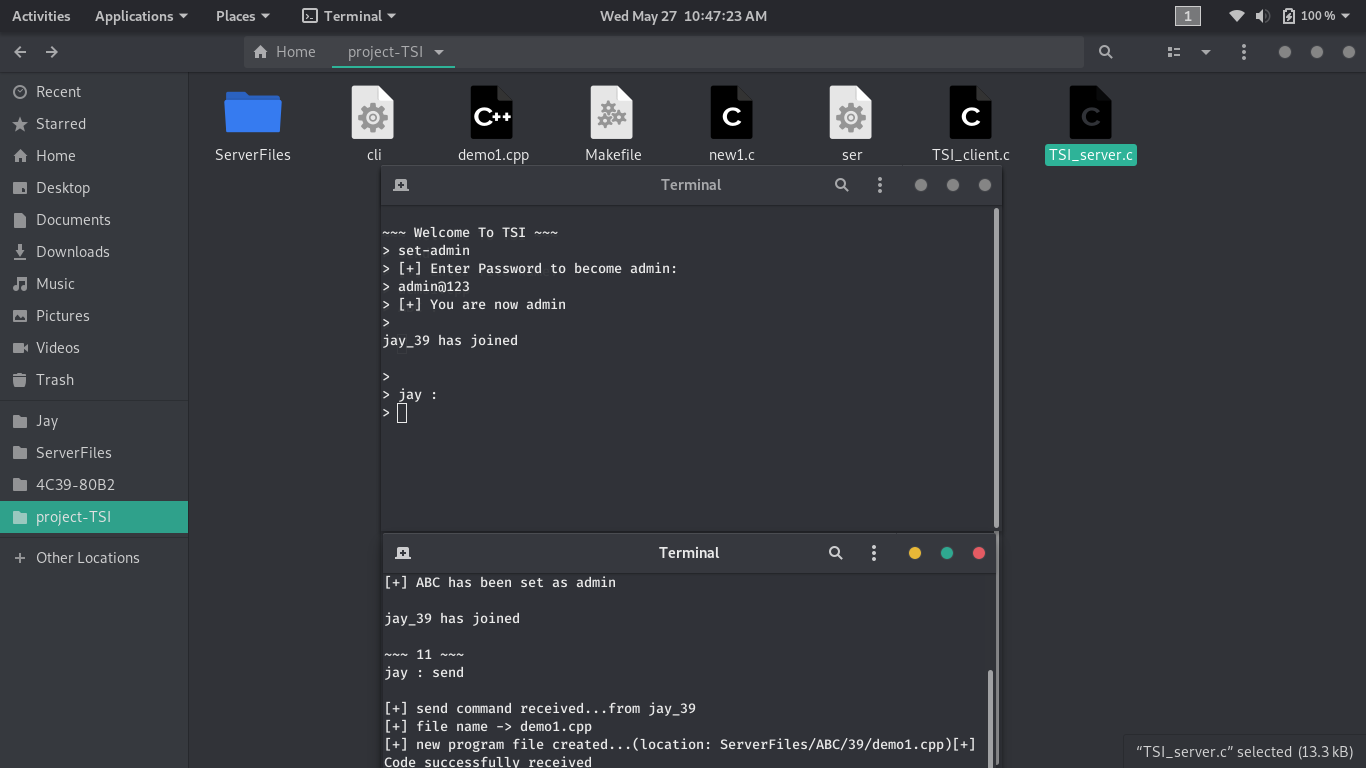
**2.File Sharing (Program here for example) :**

**File sending from Client to server :**

**Client side**

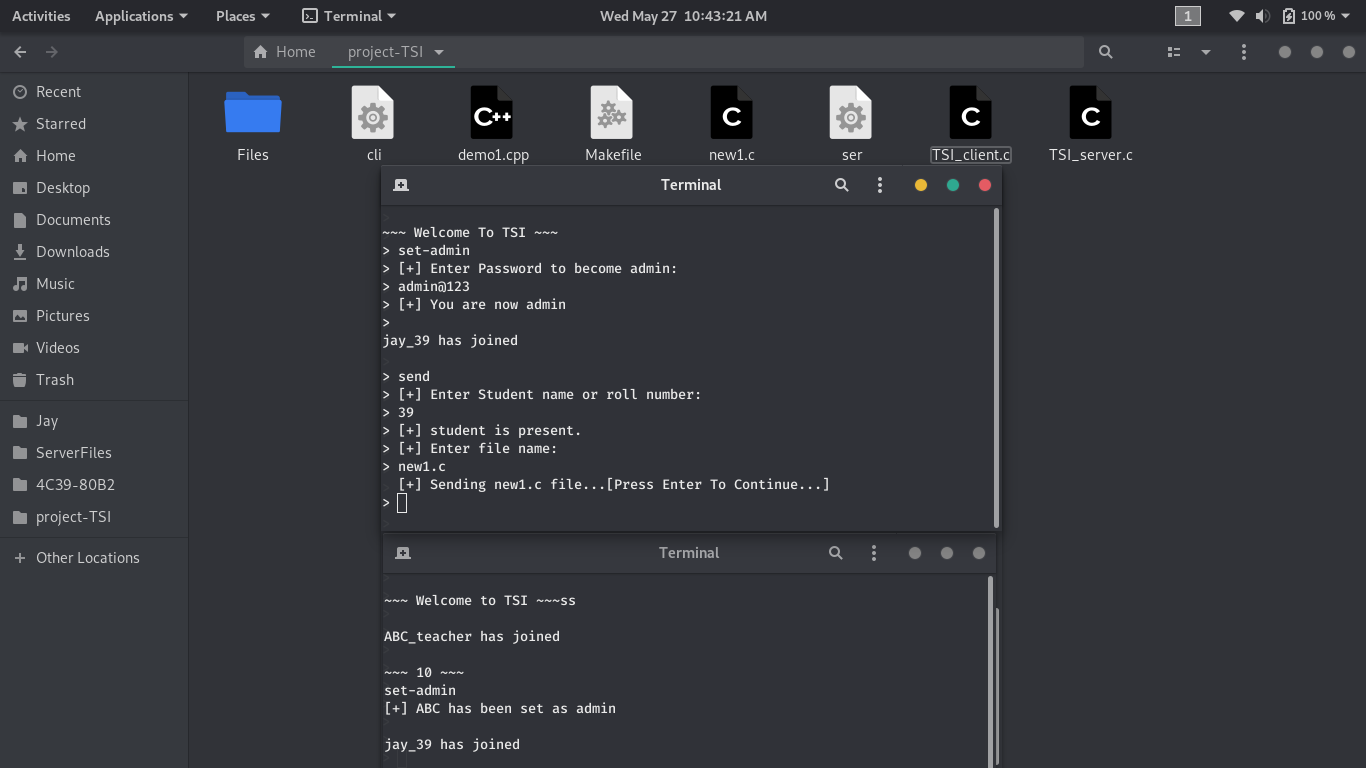
****

**Server side**

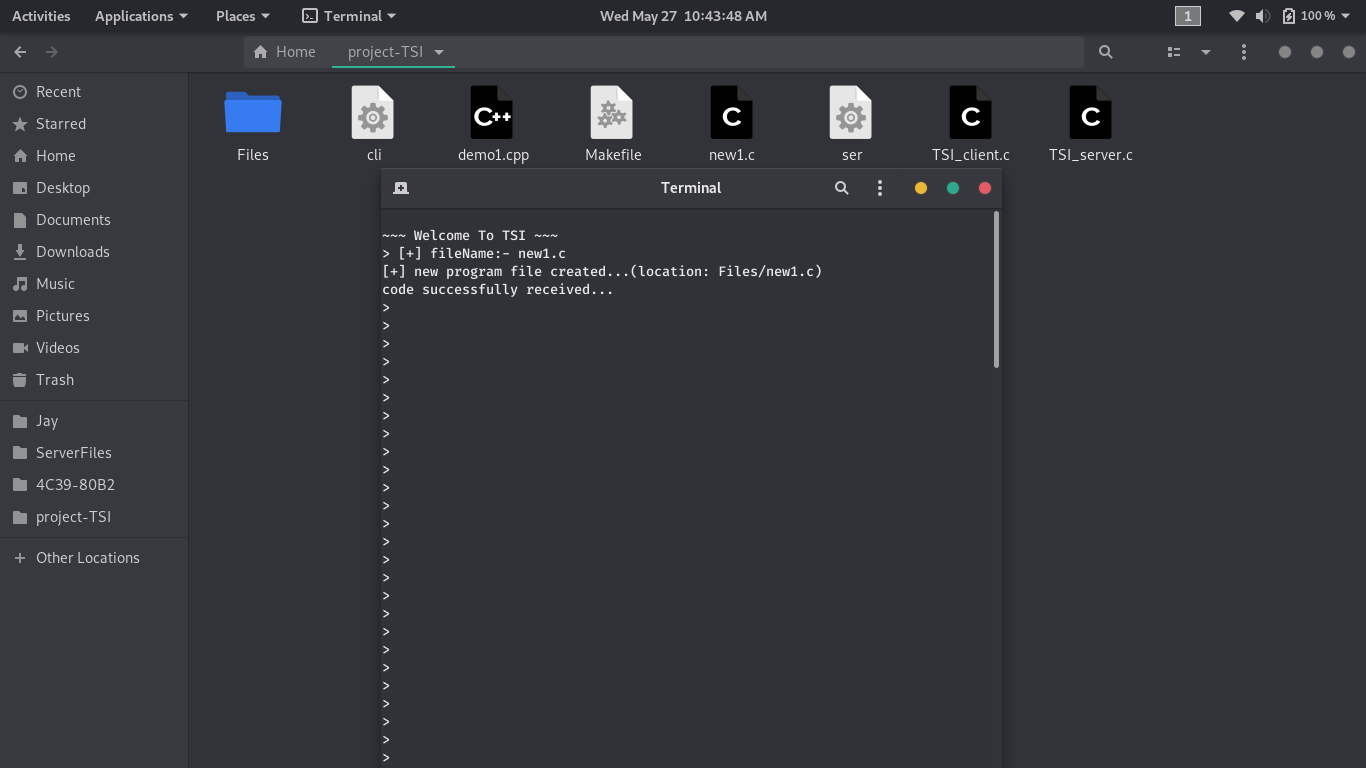
****

**File sending from server to Client :**

**Server side**

****

**Client Side**

****

**9) 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.

**10) Reference**

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. Commandline Concept - <https://www.javatpoint.com/command-line-arguments-in-c>