File Sharing Network Application

Network Assignment 1

CSC3002F

Team Members:
Zukiswa Lobola (LBLZUK002)
Simnikiwe Khonto (KHNSIM009)
Mbaliyethemba Shangase (SHNMBA004)

Due: 4 March 2020

Table of Contents

Introduction

One of the skills required from a 3rd year computer science student is familiarization with network applications as well as networking as a whole. In this assignment we were required to develop a client server file sharing application in groups of threes. File sharing refers to the transmission of a file or data through a communication channel from one computer system to another. File sharing is mediated by a communications protocol. This assignment teaches the basics of programming design and socket programming for TCP connections in Java. We learned to create a socket, bind it to a specific address as well as send and receive messages or files.

Application Design

In this assignment, we are required to design and implement a client-server file sharing application that makes use of TCP sockets. We first had to create the basic working program that connect one client to the server and transmit messages, there after configure such that it allows for the sharing of files/data. The client should be able to upload and download files (one at a time) to the server, and also to query the server for a list of files available. The network application is then upgraded to a multithreaded network to allow multiple clients to communicate with the server at the same time. As this is a network application, the clients and the server have the ability to run on different hosts.

Implementation

In this assignment the client-server architecture is used to build this file sharing networking application. The Server is crated using the ServerSocket and thread such that it can listen to multiple clients at a time. When a client is connected to the server the thread starts running and the Client uses the ServerSocket to communicate with the Server. Protocol messages are sent between the client and Server with the use of input streams as well as output stream. When the client wants to cut the communication the socket for the client is closed. The client is run with the use of a Gui as an interface for the user to interact with the application.

Protocol Specification

In this assignment, the pattern of communication will be client-server-based, meaning that a server will be responsible for the overall control and coordination of the file/data sharing process. The pattern of communication also specifies if transmission mode for files/data is unicast, multicast, or broadcast. The transmission modes may be used in different aspects of the protocol. We were required to use file/data transfer in this application protocol design.

The protocol that has been set up accepts data before it is received. Before the client can upload, download or query for files/data the server need to be started first. When the server is started it creates a socket (ServerSocket) that the clients can use to connect to the server with. It starts the thread that keeps listening for any client that wants to connect to the server. All file transferring happens via the server.

File Sharing Protocol

The client files is operated using the GUI which is initiated by the client connecting to the running server. When this is done the client starts a thread to listen to the command prompts. The GUI screen then offers the client option to upload, download and query files from the

server. When the user wants to upload files/data the client creates a pop-up screen to show available documents and runs the correct protocol to save the file on the server. When the user wants to download the file/data the user inputs the filename and the client retrieves the file from the server. When the client wants to query the server the client sends the request to the server for the list of all the files on the system and prints it back on the GUI screen.

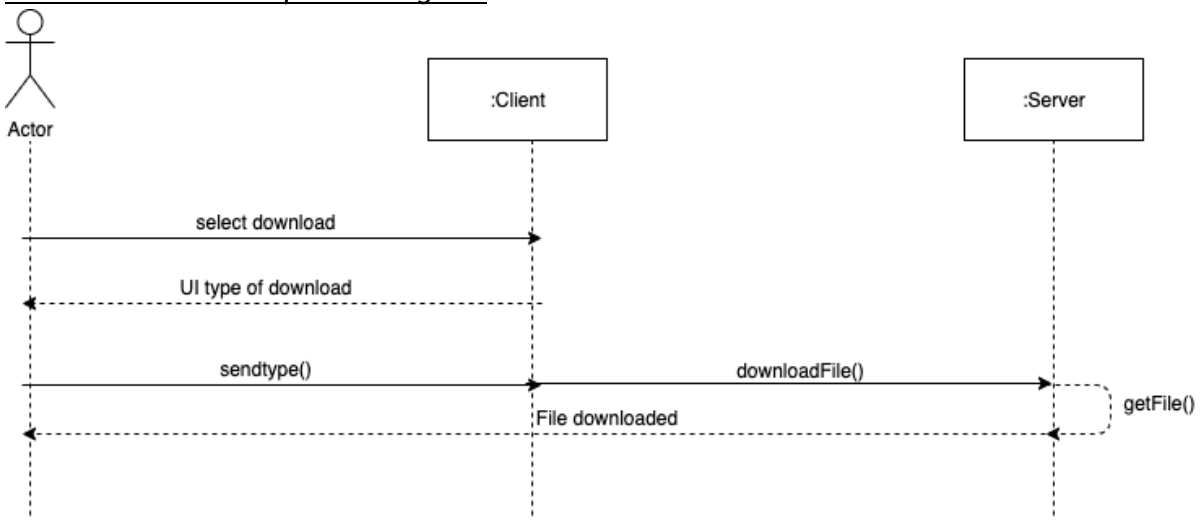
The messages being passed between the Server and Client are file based meaning that they consist of readable character strings. Text-based protocols have the advantages of being human readable, hence provide for easier understanding, monitoring and testing. In this assignment we used test-based messaging in this application protocol design. The framework of communication used in this application is real time based meaning that the client and the server transmit messages throughout the transfer process. The file sharing networking application we designed takes into consideration privacy setting by allow some files to be private available only to the user and the file permission set that some files are only read only.

Three types of messages were defined in this application, commands, data transfer, and control. The command messages constitute to the prompts printed out for the user on how to use the system. Data transfer message constitutes to the transferring of the file from the client to the server or vice versa. Control messages in this assignment are the messages that constitutes to dialogue between the client and the server on the running of the system.

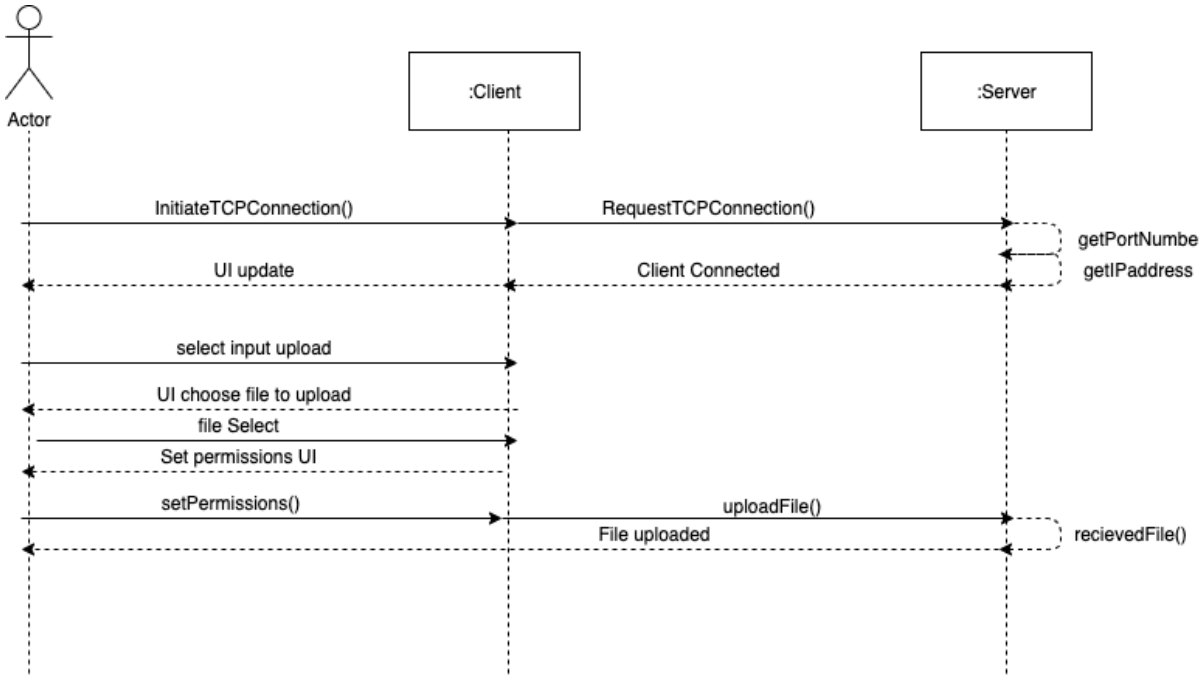
The message/file structure used in this assignment had the header and the body. The header is known by the receiver meaning that when it downloading a file from the server the client knows the header and when the client is uploading the file to the server, the server knows the header. The header describes the actual data in the message. In this case the header will alert the receiver that the data is coming from a file transfer. The body of the message will hold the file actually being transfer to the server from the client or vice versa.

The communication rules used in this application makes use of switch statements. Methods for uploading, downloading and querying files on the application are clearly written and are called when the user prompts the client on what they want to do. The communication protocol and rules can be found in the sequence diagrams below showing the uploading and downloading process.

Download Process Sequence Diagram



Upload Process Sequence Diagram



Features

When the server is first run, it keeps an open thread that keeps accepting connections from clients. When the client socket is running and connected to the server a GUI screen pop ups to prompt the user on how to user the File Sharing Networking Application. The client selects the option they want to use (download, upload and query). The server accepts the user option and makes the proper protocol response.

The list of features we have included for our File Sharing Network Application are as follows:

1. The Client is able to upload file to the Server.
    - This functionality is allowed by the Client class, it gives the user a pop up screen where the user can search for the file they want to upload, and selected to upload to the server. The file is then saved on the server.
    - The restriction added to this feature is that a client is allowed to upload one file at a time which is stored on the Server.
    - This helps the user keep the file on the server and allow for sharing with other clients that are connected to the server, as well as have a copy of the file saved on the Server as a back-up for the original.

2. The Client is able to download file from the Server.
    - This functionality is allowed by the Client class, it prompts the user to enter the filename of the file they want to download from the Server. The file is then retrieved from the Server and store within the client's files on the devices which the client is using to connect to the network.
    - The restriction added to this feature is that the client is allowed to download on file at a time from the Server and stored on the Client system.
    - This helps the user retrieve files from the server, possibly be able to read files upload by other clients, share file with different clients on the system, as well as update information on previous stored files.

3. The Client is able to query the list of files on the Server.
    - This functionality is allowed by the Client class, the client sends the query to the Server and the Server responds with printing a list of files saved on the Server.
    - The restriction added to this feature is that it only prints files that are not hidden on the server. Only the public are displayed on the list for sharing.
    - This helps the user find the file they wants to download and check whether they have upload the files correctly. The user can use this query option before downloading to check whether the file exist on the system. The user can use query option to make sure not to name files in the same way.

4. The Client is able to log off the system.
    - This functionality is allowed by the Client class, the client exits the Server by closing the socket and cutting the thread and severe the connection between the server and the client.
    - The restriction added to this feature is that the client cannot connect to the Server and perform functions when the socket is closed, and the client is no longer connected to the server.
    - This helps control the flow of traffic within the server and not run multiple unnecessary threads which requires a lot of memory.

5. The Client is able to change the file permissions.
    - This functionality is allowed by the Client class, the client makes a pop up screen that asks the user to set the file permission, setting the Executable, Writable and Readable of the file.
    - The restriction added to this feature is that the owner of the file determines how the file exist and is used in the application
    - This helps in allowing privacy setting in the file sharing networking application.

Constraints

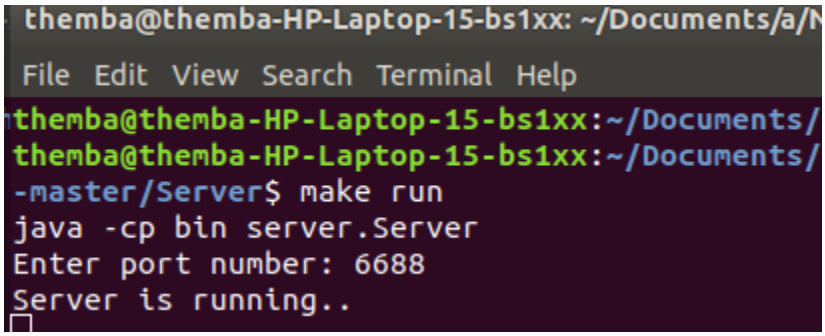
The constraints that are included in this file are as follows:

- The client can only perform its functions when it is connected to the server. This helps the system create separate threads and perform the TCP handshake.
- The client is allowed to perform on function (upload, download and query) at a time. This ensures that the system is intuitive and easy to use without confusing the files.
- The client can only access one single file at a time for either downloading and uploading, this allows for the system to be able to transfer files with the simplicity of a one to one connection.
- The server makes sure to cancel uncompleted request and inform the user that those request were cancelled. This helps the system free up memory space and make it easier to run.
- The client is allowed to set the file permission (setExecutable, setWriteable, setReadable) this allows for the privacy of files. This allows the client to save files they owned elsewhere and not available to the public.

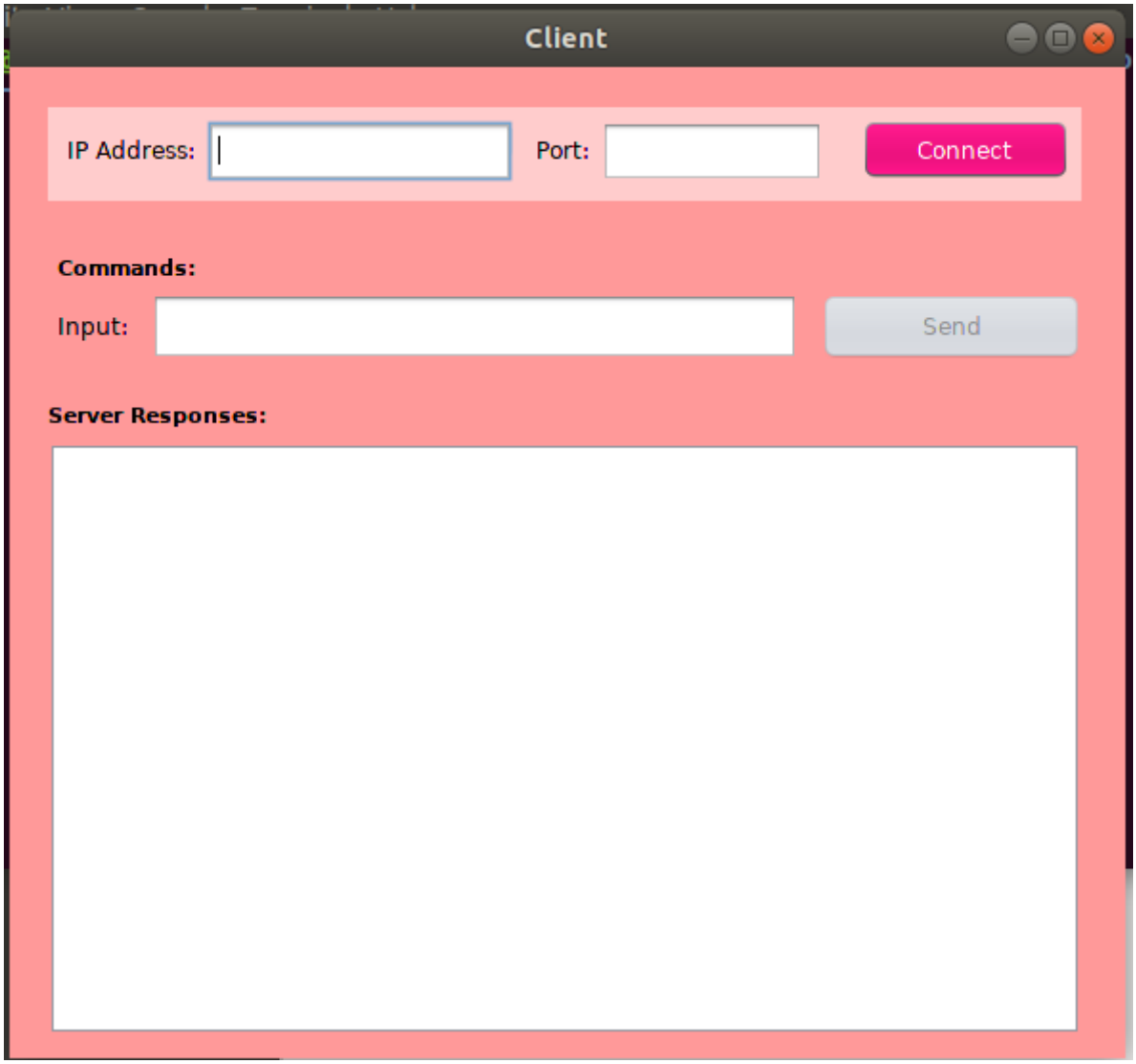
- The set file permission also allows for the public files to be treated as a read-only file for clients other the one who uploaded it.

Screenshots of how the application works

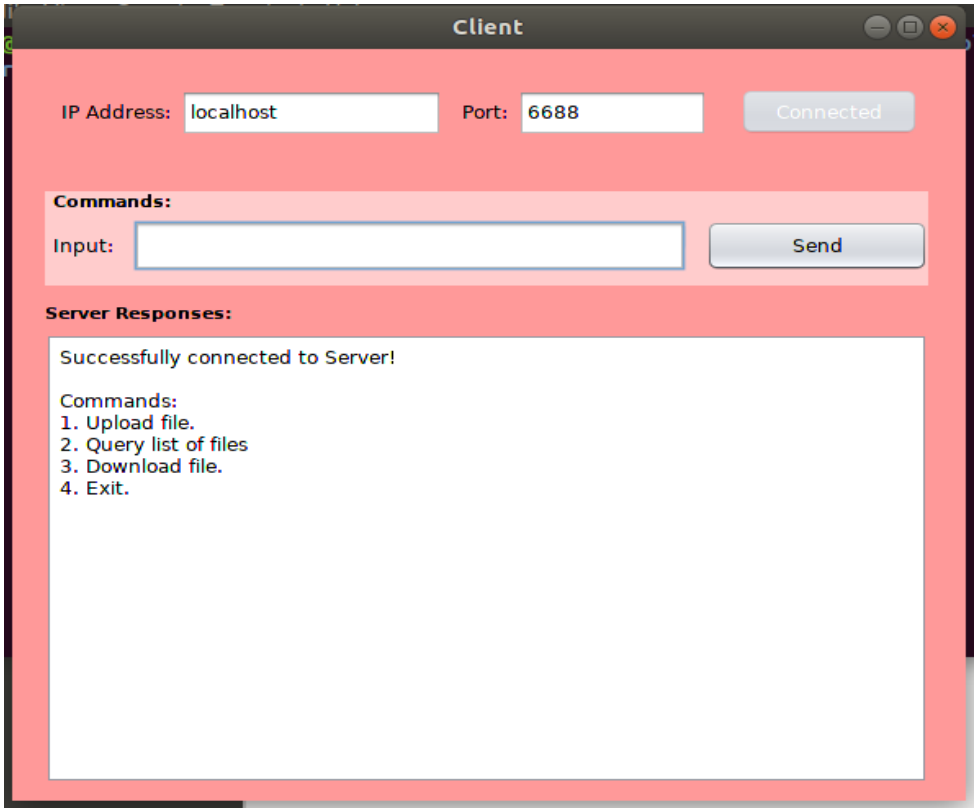
1. Initialising the server and the port



2. Runs the client



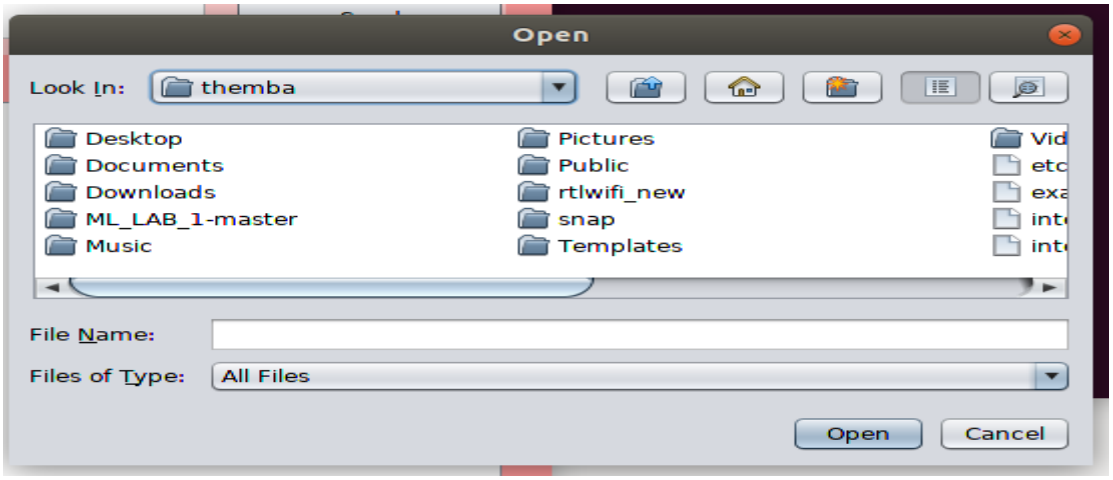
3. Connect the client to the server



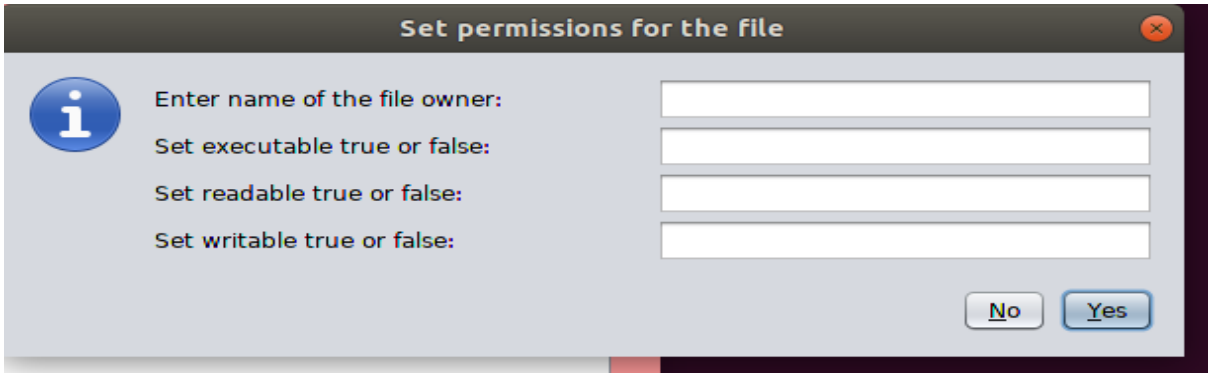
4. The server logs the client on the server.

```
themba@themba-HP-Laptop-15-bs1xx:~/Documents/a/Networks-File-Sharing-App
themba@themba-HP-Laptop-15-bs1xx:~/Documents/a/Networks-File-Sharing-App
-master/Server$ make run
java -cp bin server.Server
Enter port number: 6688
Server is running..
Accepted connection: Socket[addr=/127.0.0.1,port=48900,localport=6688]
```

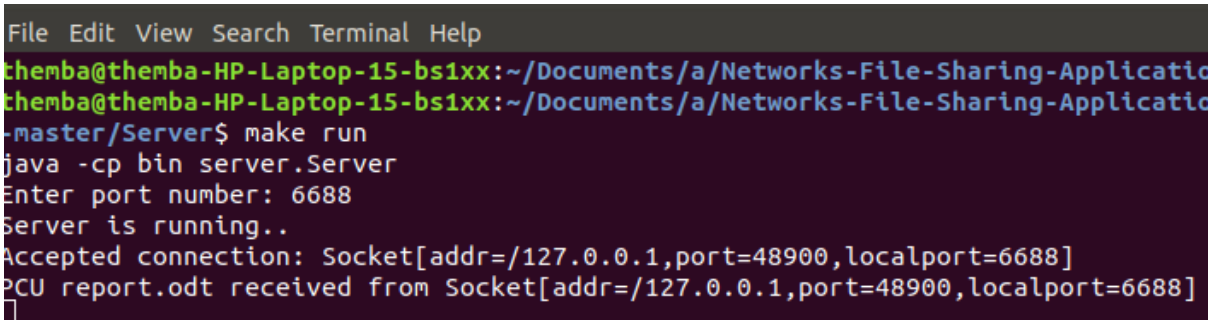
5. Pop up screen when the client wants to upload the file, to all directories and select the file



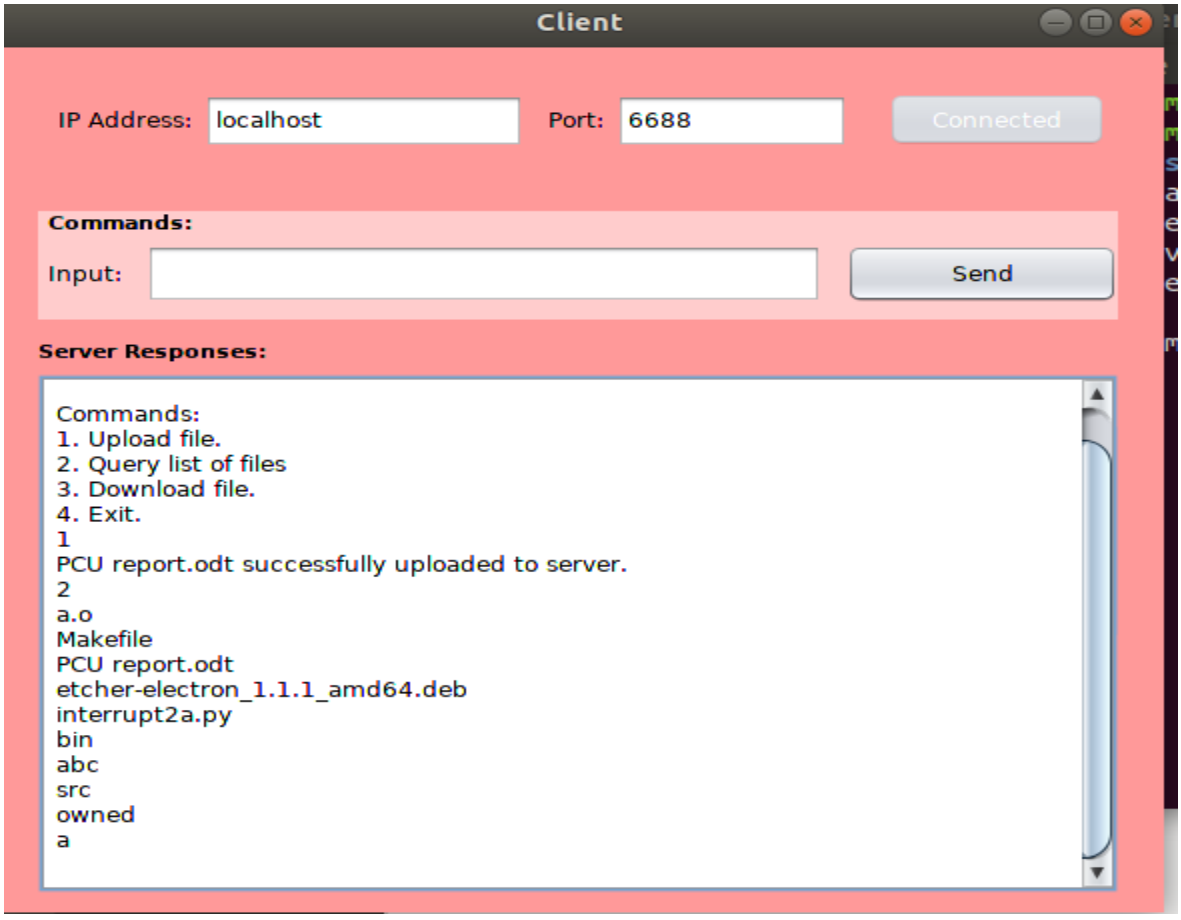
6. Set file permissions pop up



7. Shows on the server that the file is uploaded.



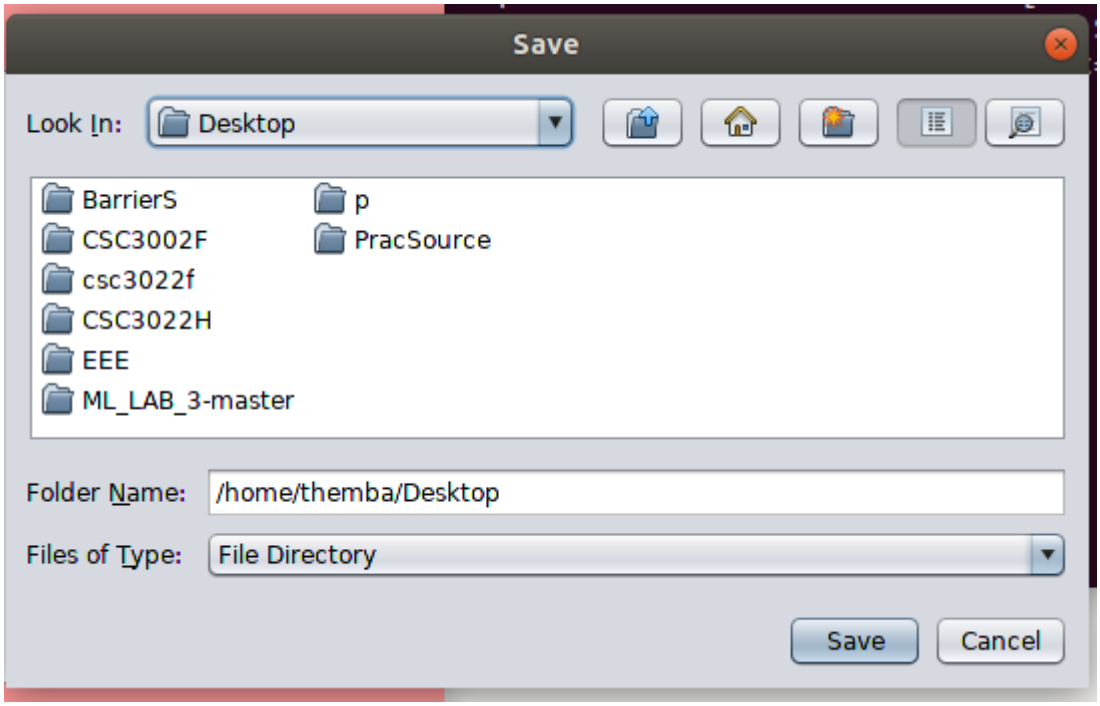
8. Prints out the list when the server is queried by put 2 in the input text field



9. The server retrieves the file path.

```
File  Edit  View  Search  Terminal  Help
themba@themba-HP-Laptop-15-bs1xx:~/Documents/a/Networks-File-Sharing-Applicatio
themba@themba-HP-Laptop-15-bs1xx:~/Documents/a/Networks-File-Sharing-Applicatio
-master/Server$ make run
java -cp bin server.Server
Enter port number: 6688
Server is running..
Accepted connection: Socket[addr=/127.0.0.1,port=48900,localport=6688]
PCU report.odt received from Socket[addr=/127.0.0.1,port=48900,localport=6688]
/home/themba/Documents/a/Networks-File-Sharing-Application-master/Server
```

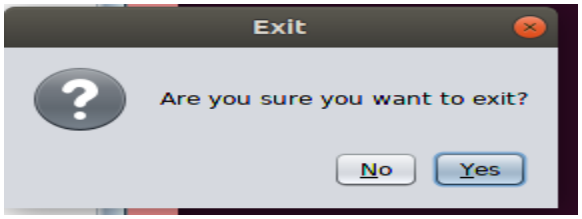
10. Pop up screen the client is downloading a file from the server.



11. The server shows the message that the file has been downloaded.

```
File  Edit  View  Search  Terminal  Help
themba@themba-HP-Laptop-15-bs1xx:~/Documents/a/Networks-File-Sharing-Application
themba@themba-HP-Laptop-15-bs1xx:~/Documents/a/Networks-File-Sharing-Application
-master/Server$ make run
java -cp bin server.Server
Enter port number: 6688
Server is running..
Accepted connection: Socket[addr=/127.0.0.1,port=48900,localport=6688]
PCU report.odt received from Socket[addr=/127.0.0.1,port=48900,localport=6688]
/home/themba/Documents/a/Networks-File-Sharing-Application-master/Server
abc sent to Socket[addr=/127.0.0.1,port=48900,localport=6688].
```

12. The pop screen when you want to exit the Client Gui.



13. The server logs of the client when it is exited.