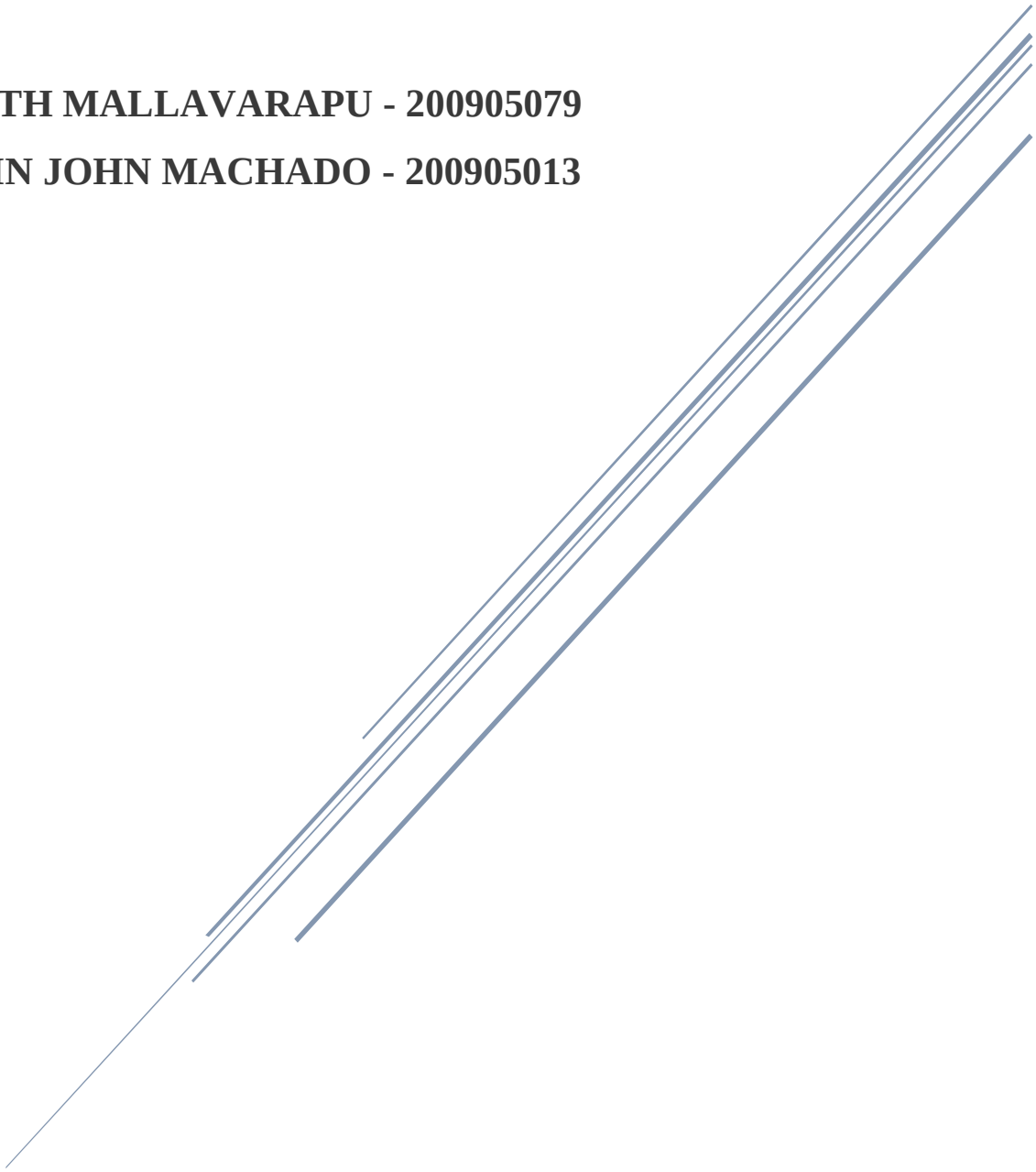
FILE SERVER SYSTEM

LIKHITH MALLAVARAPU - 200905079

CALVIN JOHN MACHADO - 200905013



Department of Computer Science, Manipal Institute of Technology, Manipal, Karnataka, India

ABSTRACT

The project is a file server system which is implemented in the form of a client server model. One server with multiple clients functioning concurrently is possible. The client can connect to the server using a TCP connection which happens automatically as soon as the files are executed. The features also include a GUI application allowing an interactive experience. The files can be selected on the client side which is then sent to the server side where it can be stored and downloaded on the server side; multiple clients can send files to the server which can then be downloaded on the server side. The types of files that can be sent include images in the form of .jpg, .jpeg, .png etc, text files, .c files, .py files. Java files etc.

INTRODUCTION

GENERAL INTRODUCTION

The central server in a computer network that is responsible for the storage and management of data files is called a File Server. In a File Server, users access a central storage space that acts as a medium to store the internal data. The users can share information over a network without having to physically transfer files. Its advantages include –

1) Helps in resource and information sharing.
2) Helps in central storage of data.
3) Helps in connecting with multiple computers for sending and receiving information when accessing the network.
4) Faster-problem-solving.
5) Boots Storage Capacity.
6) Highly flexible and reliable.

HARDWARE AND SOFTWARE REQUIREMENTS

Hardware components

Servers - Servers are high-configuration computers that manage the resources of the network. The network operating system is typically installed in the server and so they give user accesses to the network resources. Servers can be of various kinds: file servers, database servers, print servers etc.

Clients - Clients are computers that request and receive service from the servers to access and use the network resources.

Transmission Media - Transmission media are the channels through which data is transferred from one device to another in a network. Transmission media may be guided media like coaxial cable, fibre optic cables etc; or maybe unguided media like microwaves, infra-red waves etc.

Connecting Devices - Connecting devices act as middleware between networks or computers, by binding the network media together. Some of the common connecting devices are Routers, Repeaters and Switches.

Network Interface Card - Also known as network adapter, interfaces a computer board with the network medium.

Software Components

Networking Operating System - Network Operating Systems is typically installed in the server and facilitate workstations in a network to share files, database, applications, printers etc.

Protocol Suite - A protocol is a rule or guideline followed by each computer for data communication. Protocol suite is a set of related protocols that are laid down for computer networks. The two popular protocol suites are OSI Model (Open System Interconnections) and TCP / IP Model.

Network Driver Software Provides an interface between the high level networking software and the particular Network Interface Card (NIC) that is being used for physical LAN communication.

TCP/IP Transmission Control Protocol/Internet Protocol, the suite of communications protocols used to connect hosts on the Internet. TCP/IP uses several protocols, the two main ones being TCP and IP. TCP/IP is built into the UNIX operating system and is used by the Internet, making it the de facto standard for transmitting data over networks.

The latest version of Java must be installed.

PROBLEM DEFINITION

The problem was to build an application that allows an easy way for multiple users to share different kinds of files to a common server concurrently which can then be downloaded on the servers side and stored in the local PC.

OBJECTIVES

Following are the objectives of this project –

1) To make it convenient for users to send, store and download files.

2) To make the application easy and understanble to use.

3) To demonstrate the reliable data transfer features of TCP / IP protocol.

4) File servers provide a central location to store all of your business files so that multiple users can easily work with the same documents, spreadsheets and other data.

5) A file server is a great option for smaller businesses that want everything to be stored locally, while still allowing multiple employees access to various business files without having to physically access the device.

6) Advantage of having your own file server is the ability to configure it in as many ways as you wish. This gives you options about how you want files shared, how credentials will be assigned, where people will be able to access files from, etc. This will give you flexibility and allows you to set things up in a way that is unique, secure, and specialized for your organization.

METHODOLOGY

Initially the connection between the client and server has to happen which is done using the built in Java functions using an IP address and a port number.

Once this TCP connection is established, the client is prompted to select one or more files to be transferred onto the server side. Once on the server side a list is maintained of all the files currently on the server which can then be downloaded on the server side and stored for various purposes. This process can be repeated with multiple clients concurrently. The whole application is coded in Java and to make it easier for users to use, a GUI for the application is provided which is built using the Java Swing framework and the connection is established directly making it convenient for any user to use.

IMPLEMENTATION

```
//Server.java

import javax.swing.*;
import javax.swing.border.EmptyBorder;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.MouseEvent;
import java.awt.event.MouseListener;
import java.io.*;
import java.net.ServerSocket;
import java.net.Socket;
import java.util.ArrayList;
public class Server {
    static ArrayList<MyFile> myFiles = new ArrayList<>();
    public static void main(String[] args) throws IOException {
        int fileId = 0;
        JFrame jFrame = new JFrame("Server");
        jFrame.setSize(400, 400);
        jFrame.setLayout(new BoxLayout(jFrame.getContentPane(),
BoxLayout.Y_AXIS));
        jFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JPanel jPanel = new JPanel();
        jPanel.setLayout(new BoxLayout(jPanel, BoxLayout.Y_AXIS));
        JScrollPane jScrollPane = new JScrollPane(jPanel);
jScrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR
_ALWAYS);
```

```
        JLabel jlTitle = new JLabel("File Receiver");
        jlTitle.setFont(new Font("Arial", Font.BOLD, 25));
        jlTitle.setBorder(new EmptyBorder(20,0,10,0));
        jlTitle.setAlignmentX(Component.CENTER_ALIGNMENT);
        jFrame.add(jlTitle);
        jFrame.add(jScrollPane);
        jFrame.setVisible(true);
        ServerSocket serverSocket = new ServerSocket(1234);
        while (true) {
            try {
                Socket socket = serverSocket.accept();
                DataInputStream dataInputStream = new
DataInputStream(socket.getInputStream());
                int fileNameLength = dataInputStream.readInt();
                if (fileNameLength > 0) {
                byte[] fileNameBytes = new byte[fileNameLength];
                    dataInputStream.readFully(fileNameBytes, 0,
fileNameBytes.length);
                    String fileName = new String(fileNameBytes);
                    int fileContentLength = dataInputStream.readInt();
                    if (fileContentLength > 0) {
                        byte[] fileContentBytes = new byte[fileContentLength];
                        dataInputStream.readFully(fileContentBytes, 0,
fileContentBytes.length);
                        JPanel jpFileRow = new JPanel();
                        jpFileRow.setLayout(new BoxLayout(jpFileRow,
BoxLayout.X_AXIS));
                        JLabel jlFileName = new JLabel(fileName);
                        jlFileName.setFont(new Font("Arial", Font.BOLD, 20));
                        jlFileName.setBorder(new EmptyBorder(10,0, 10,0));
```

```
                    if (getFileExtension(fileName).equalsIgnoreCase("txt")) {
                        jpFileRow.setName((String.valueOf(fileId)));
                        jpFileRow.addMouseListener(getMyMouseListener());
                        jpFileRow.add(jlFileName);
                        jPanel.add(jpFileRow);
                        jFrame.validate();
                    } else {
                        jpFileRow.setName((String.valueOf(fileId)));
                        jpFileRow.addMouseListener(getMyMouseListener());
                        jpFileRow.add(jlFileName);
                        jPanel.add(jpFileRow);
                        jFrame.validate();
                    }
                    myFiles.add(new MyFile(fileId, fileName, fileContentBytes,
getFileExtension(fileName)));
                    fileId++;
                }
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
public static String getFileExtension(String fileName) {
    int i = fileName.lastIndexOf('.');
    if (i > 0) {
        return fileName.substring(i + 1);
    } else {
        return "No extension found.";
```

```
        }
    }
    public static MouseListener getMyMouseListener() {
        return new MouseListener() {
            @Override
            public void mouseClicked(MouseEvent e) {
                JPanel jPanel = (JPanel) e.getSource();
                int fileId = Integer.parseInt(jPanel.getName());
                for (MyFile myFile : myFiles) {
                    if (myFile.getId() == fileId) {
                        JFrame jfPreview = createFrame(myFile.getName(),
myFile.getData(), myFile.getFileExtension());
                        jfPreview.setVisible(true);
                    }
                }
            }
            @Override
            public void mousePressed(MouseEvent e) {
            }
            @Override
            public void mouseReleased(MouseEvent e) {
            }
            @Override
            public void mouseEntered(MouseEvent e) {
            }
            @Override
            public void mouseExited(MouseEvent e) {
            }
        };
```

```
    }
    public static JFrame createFrame(String fileName, byte[] fileData, String
fileExtension) {
        JFrame jFrame = new JFrame("File Downloader");
        jFrame.setSize(400, 400);
        JPanel jPanel = new JPanel();
        jPanel.setLayout(new BoxLayout(jPanel, BoxLayout.Y_AXIS));
        JLabel jlTitle = new JLabel("File Downloader");
        jlTitle.setAlignmentX(Component.CENTER_ALIGNMENT);
        jlTitle.setFont(new Font("Arial", Font.BOLD, 25));
        jlTitle.setBorder(new EmptyBorder(20,0,10,0));
        JLabel jlPrompt = new JLabel("Are you sure you want to download " +
fileName + "?");
        jlPrompt.setFont(new Font("Arial", Font.BOLD, 20));
        jlPrompt.setBorder(new EmptyBorder(20,0,10,0));
        jlPrompt.setAlignmentX(Component.CENTER_ALIGNMENT);
        JButton jbYes = new JButton("Yes");
        jbYes.setPreferredSize(new Dimension(150, 75));
        jbYes.setFont(new Font("Arial", Font.BOLD, 20));
        JButton jbNo = new JButton("No");
        jbNo.setPreferredSize(new Dimension(150, 75));
        jbNo.setFont(new Font("Arial", Font.BOLD, 20));
        JLabel jlFileContent = new JLabel();
        jlFileContent.setAlignmentX(Component.CENTER_ALIGNMENT);
        JPanel jpButtons = new JPanel();
        jpButtons.setBorder(new EmptyBorder(20, 0, 10, 0));
        jpButtons.add(jbYes);
        jpButtons.add(jbNo);
        if (fileExtension.equalsIgnoreCase("txt")) {
```

```
            jlFileContent.setText("<html>" + new String(fileData) + "</html>");
        } else {
            jlFileContent.setIcon(new ImageIcon(fileData));
        }.
        jbYes.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                File fileToDownload = new File(fileName);
                try {
                    FileOutputStream fileOutputStream = new
FileOutputStream(fileToDownload);
                    fileOutputStream.write(fileData);
                    fileOutputStream.close();
                    jFrame.dispose();
                } catch (IOException ex) {
                    ex.printStackTrace();
                }
            }
        });
        jbNo.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                jFrame.dispose();
            }
        });
        jPanel.add(jlTitle);
        jPanel.add(jlPrompt);
        jPanel.add(jlFileContent);
        jPanel.add(jpButtons);
```

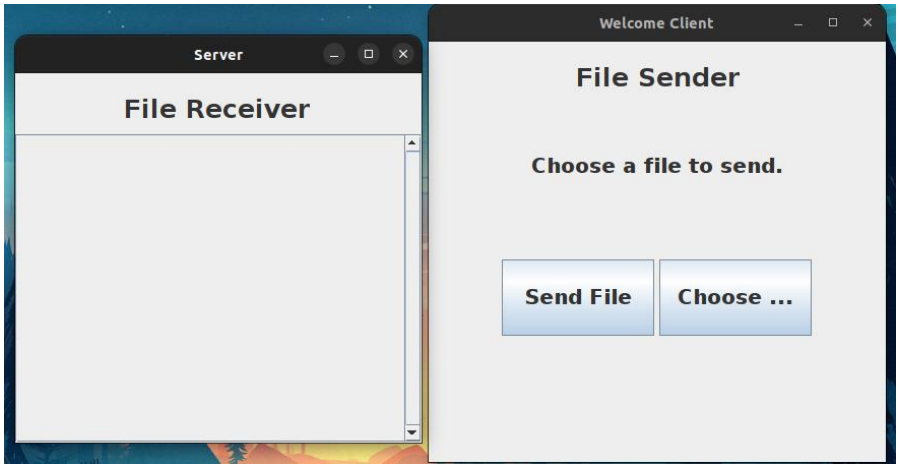
```
        jFrame.add(jPanel);
        return jFrame;
    }
}
//Client.java
import javax.swing.*;
import javax.swing.border.EmptyBorder;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.io.*;
import java.net.Socket;
public class Client {
    public static void main(String[] args) {
        final File[] fileToSend = new File[1];
        JFrame jFrame = new JFrame("Welcome Client");
        jFrame.setSize(450, 450);
        jFrame.setLayout(new BoxLayout(jFrame.getContentPane(),
BoxLayout.Y_AXIS));
        jFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel jlTitle = new JLabel("File Sender");
        jlTitle.setFont(new Font("Arial", Font.BOLD, 25));
        jlTitle.setBorder(new EmptyBorder(20,0,10,0));
```

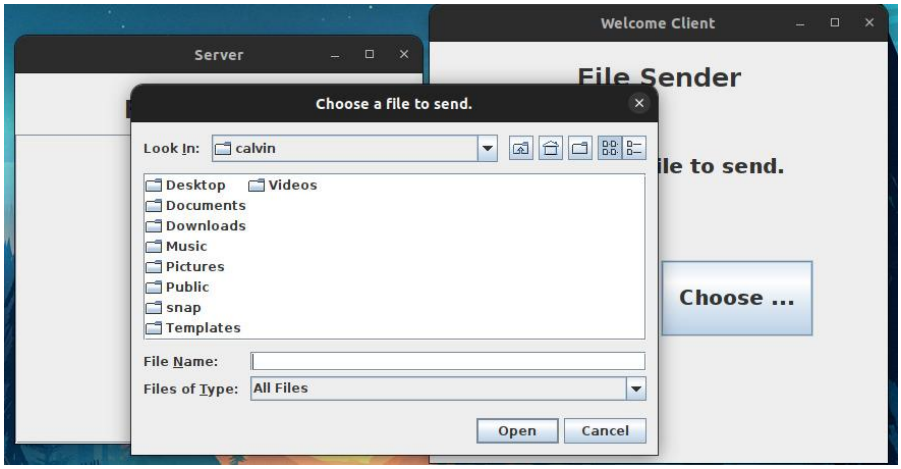
```
        jlTitle.setAlignmentX(Component.CENTER_ALIGNMENT);
        JLabel jlFileName = new JLabel("Choose a file to send.");
        jlFileName.setFont(new Font("Arial", Font.BOLD, 20));
        jlFileName.setBorder(new EmptyBorder(50, 0, 0, 0));
        jlFileName.setAlignmentX(Component.CENTER_ALIGNMENT);
        JPanel jpButton = new JPanel();
        jpButton.setBorder(new EmptyBorder(75, 0, 10, 0));
        JButton jbSendFile = new JButton("Send File");
        jbSendFile.setPreferredSize(new Dimension(150, 75));

        jbSendFile.setFont(new Font("Arial", Font.BOLD, 20));
        JButton jbChooseFile = new JButton("Choose File");
        jbChooseFile.setPreferredSize(new Dimension(150, 75));
        jbChooseFile.setFont(new Font("Arial", Font.BOLD, 20));
        jpButton.add(jbSendFile);
        jpButton.add(jbChooseFile);
        jbChooseFile.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JFileChooser jFileChooser = new JFileChooser();
                jFileChooser.setDialogTitle("Choose a file to send.");
                if (jFileChooser.showOpenDialog(null)  ==
JFileChooser.APPROVE_OPTION) {
```

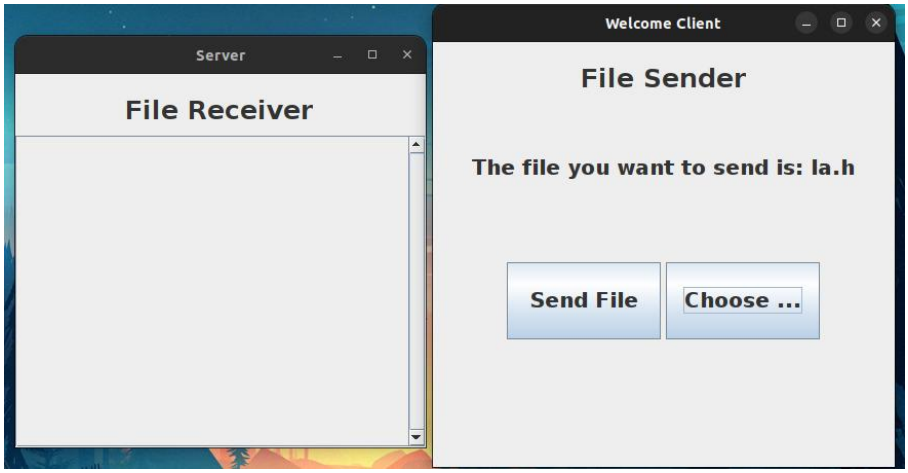
```
                fileToSend[0] = jFileChooser.getSelectedFile();

                jlFileName.setText("The file you want to send is: " +
fileToSend[0].getName());

            }

        }

    });

    jbSendFile.addActionListener(new ActionListener() {

        @Override

        public void actionPerformed(ActionEvent e) {

            if (fileToSend[0] == null) {

                jlFileName.setText("Please choose a file to send first!");

            } else {

                try {
                    FileInputStream fileInputStream = new
FileInputStream(fileToSend[0].getAbsolutePath());

                    Socket socket = new Socket("localhost", 1234);
                    DataOutputStream dataOutputStream = new
DataOutputStream(socket.getOutputStream());

                    String fileName = fileToSend[0].getName();

                    byte[] fileNameBytes = fileName.getBytes();

                    byte[] fileBytes = new byte[(int)fileToSend[0].length()];

                    fileInputStream.read(fileBytes);

                    dataOutputStream.writeInt(fileNameBytes.length);

                    dataOutputStream.write(fileNameBytes);
```

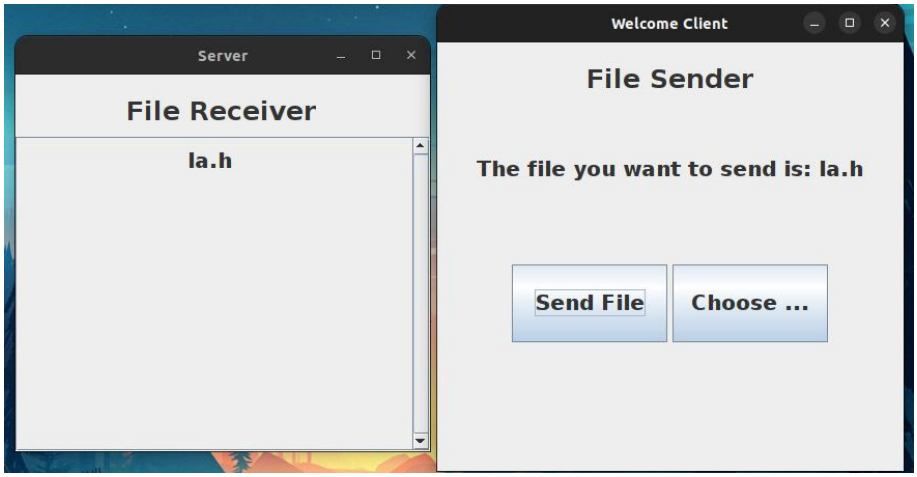
```
                    dataOutputStream.writeInt(fileBytes.length);
                    dataOutputStream.write(fileBytes);
                } catch (IOException ex) {
                    ex.printStackTrace();
                }
            }
        }
    });
    jFrame.add(jlTitle);
    jFrame.add(jlFileName);
    jFrame.add(jpButton);
    jFrame.setVisible(true);
  }
}
```

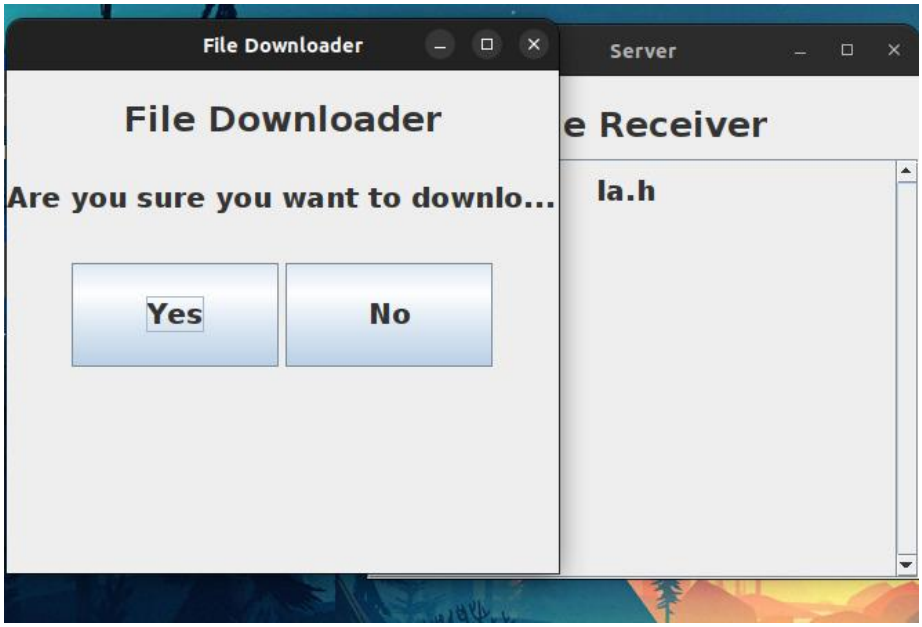
OUTPUT


Server
File Receiver
Welcome Client
File Sender
Choose a file to send.
Send File
Choose ...


Server
Welcome Client
File Sender
Choose a file to send.
Look In:
calvin
Desktop
Videos
Documents
Downloads
Music
Pictures
Public
snap
Templates
File Name:
Files of Type:
All Files
Open
Cancel
Choose ...


Server
File Receiver
Welcome Client
File Sender
The file you want to send is: la.h
Send File
Choose ...





CONTRIBUTION SUMMARY

Likhith contributed by doing the client side of the file server system and helping with the report writing for the project.

Calvin contributed by making the server side of the system along with the GUI and integrating it with both the client and server side, along with writing majority portion of the report.

REFERENCES

1) https://www.javatpoint.com/java-swing

2) https://www.geeksforgeeks.org/what-is-file-server/

3) https://www.bemopro.com/cybersecurity-blog/what-is-a-file-server#:~:text=File%20servers%20provide%20a%20central,sure%20their%20data%20is%20safe.

4) https://slideplayer.com/slide/3837060/