NAME: GVS SAI MADHAV
REG.NO: 19BCN7228
LAB-6: Java Socket Client And Server File Transfer:

Codes:

Client.java:

```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) {

        // Accessed from within inner class needs to be final or effectively
final.
        final File[] fileToSend = new File[1];

        // Set the frame to house everything.
        JFrame jFrame = new JFrame("Java Client");
        // Set the size of the frame.
        jFrame.setSize(450, 450);
        // Make the layout to be box layout that places its children on top of
each other.
        jFrame.setLayout(new BoxLayout(jFrame.getContentPane(),
BoxLayout.Y_AXIS));
        // Make it so when the frame is closed the program exits successfully.
        jFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Title above panel.
        JLabel jlTitle = new JLabel("Java File Sender");
        // Change the font family, size, and style.
        jlTitle.setFont(new Font("Arial", Font.BOLD, 25));
        // Add a border around the label for spacing.
        jlTitle.setBorder(new EmptyBorder(20,0,10,0));
        // Make it so the title is centered horizontally.
        jlTitle.setAlignmentX(Component.CENTER_ALIGNMENT);

        // Label that has the file name.
        JLabel jlFileName = new JLabel("Choose a file to send.");
        // Change the font.
        jlFileName.setFont(new Font("Arial", Font.BOLD, 20));
        // Make a border for spacing.
        jlFileName.setBorder(new EmptyBorder(50, 0, 0, 0));
        // Center the label on the x axis (horizontally).
        jlFileName.setAlignmentX(Component.CENTER_ALIGNMENT);

        // Panel that contains the buttons.
        JPanel jpButton = new JPanel();
        // Border for panel that houses buttons.
        jpButton.setBorder(new EmptyBorder(75, 0, 10, 0));
        // Create send file button.
        JButton jbSendFile = new JButton("Send File");
        // Set preferred size works for layout containers.
        jbSendFile.setPreferredSize(new Dimension(150, 75));
        // Change the font style, type, and size for the button.
        jbSendFile.setFont(new Font("Arial", Font.BOLD, 20));
        // Make the second button to choose a file.
        JButton jbChooseFile = new JButton("Choose File");
        // Set the size which must be preferred size for within a container.
        jbChooseFile.setPreferredSize(new Dimension(150, 75));
        // Set the font for the button.
        jbChooseFile.setFont(new Font("Arial", Font.BOLD, 20));

        // Add the buttons to the panel.
        jpButton.add(jbSendFile);
        jpButton.add(jbChooseFile);

        // Button action for choosing the file.
        // This is an inner class so we need the fileToSend to be final.
        jbChooseFile.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Create a file chooser to open the dialog to choose a file.
                JFileChooser jFileChooser = new JFileChooser();
                // Set the title of the dialog.
                jFileChooser.setDialogTitle("Choose a file to send.");
                // Show the dialog and if a file is chosen from the file chooser
execute the following statements.
                if (jFileChooser.showOpenDialog(null)  ==
JFileChooser.APPROVE_OPTION) {
                    // Get the selected file.
                    fileToSend[0] = jFileChooser.getSelectedFile();
                    // Change the text of the java swing label to have the file
name.
                    jlFileName.setText("The file you want to send is: " +
fileToSend[0].getName());
                }
            }
        });


        // Sends the file when the button is clicked.
        jbSendFile.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // If a file has not yet been selected then display this message.
                if (fileToSend[0] == null) {
                    jlFileName.setText("Please choose a file to send first!");
                    // If a file has been selected then do the following.
                } else {
                    try {
                        // Create an input stream into the file you want to send.
                        FileInputStream fileInputStream = new
FileInputStream(fileToSend[0].getAbsolutePath());
                        // Create a socket connection to connect with the server.
                        Socket socket = new Socket("localhost", 1234);
                        // Create an output stream to write to write to the
server over the socket connection.
                        DataOutputStream dataOutputStream = new
DataOutputStream(socket.getOutputStream());
                        // Get the name of the file you want to send and store it
in filename.
                        String fileName = fileToSend[0].getName();
                        // Convert the name of the file into an array of bytes to
be sent to the server.
                        byte[] fileNameBytes = fileName.getBytes();
                        // Create a byte array the size of the file so don't send
too little or too much data to the server.
                        byte[] fileBytes = new byte[(int)fileToSend[0].length()];
                        // Put the contents of the file into the array of bytes
to be sent so these bytes can be sent to the server.
                        fileInputStream.read(fileBytes);
                        // Send the length of the name of the file so server
knows when to stop reading.
                        dataOutputStream.writeInt(fileNameBytes.length);
                        // Send the file name.
                        dataOutputStream.write(fileNameBytes);
                        // Send the length of the byte array so the server knows
when to stop reading.
                        dataOutputStream.writeInt(fileBytes.length);
                        // Send the actual file.
                        dataOutputStream.write(fileBytes);
                    } catch (IOException ex) {
                        ex.printStackTrace();
                    }
                }
            }
        });

        // Add everything to the frame and make it visible.
        jFrame.add(jlTitle);
        jFrame.add(jlFileName);
        jFrame.add(jpButton);
        jFrame.setVisible(true);
    }

}
```

Server.java:

```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 {

    // Array list to hold information about the files received.
    static ArrayList<MyFile> myFiles = new ArrayList<>();

    public static void main(String[] args) throws IOException {

        // Used to track the file (jpanel that has the file name in it on a
label).
        int fileId = 0;

        // Main container, set the name.
        JFrame jFrame = new JFrame("Java File Server");
        // Set the size of the frame.
        jFrame.setSize(400, 400);
        // Give the frame a box layout that stacks its children on top of each
other.
        jFrame.setLayout(new BoxLayout(jFrame.getContentPane(),
BoxLayout.Y_AXIS));
        // When closing the frame also close the program.
        jFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

        // Panel that will hold the title label and the other jpanels.
        JPanel jPanel = new JPanel();
        // Make the panel that contains everything to stack its child elements on
top of eachother.
        jPanel.setLayout(new BoxLayout(jPanel, BoxLayout.Y_AXIS));

        // Make it scrollable when the data gets in jpanel.
        JScrollPane jScrollPane = new JScrollPane(jPanel);
        // Make it so there is always a vertical scrollbar.
        jScrollPane.setVerticalScrollBarPolicy(JScrollPane.VERTICAL_SCROLLBAR_ALW
AYS);

        // Title above panel.
        JLabel jlTitle = new JLabel("Java File Receiver");
        // Change the font of the title.
        jlTitle.setFont(new Font("Arial", Font.BOLD, 25));
        // Add a border around the title for spacing.
        jlTitle.setBorder(new EmptyBorder(20,0,10,0));
        // Center the title horizontally in the middle of the frame.
        jlTitle.setAlignmentX(Component.CENTER_ALIGNMENT);

        // Add everything to the main GUI.
        jFrame.add(jlTitle);
        jFrame.add(jScrollPane);
        // Make the GUI show up.
        jFrame.setVisible(true);

        // Create a server socket that the server will be listening with.
        ServerSocket serverSocket = new ServerSocket(1234);

        // This while loop will run forever so the server will never stop unless
the application is closed.
        while (true) {

            try {
                // Wait for a client to connect and when they do create a socket
to communicate with them.
                Socket socket = serverSocket.accept();

                // Stream to receive data from the client through the socket.
                DataInputStream dataInputStream = new
DataInputStream(socket.getInputStream());

                // Read the size of the file name so know when to stop reading.
                int fileNameLength = dataInputStream.readInt();
                // If the file exists
                if (fileNameLength > 0) {
                    // Byte array to hold name of file.
                    byte[] fileNameBytes = new byte[fileNameLength];
                    // Read from the input stream into the byte array.
                    dataInputStream.readFully(fileNameBytes, 0,
fileNameBytes.length);
                    // Create the file name from the byte array.
                    String fileName = new String(fileNameBytes);
```

```java
                    // Read how much data to expect for the actual content of the
file.
                    int fileContentLength = dataInputStream.readInt();
                    // If the file exists.
                    if (fileContentLength > 0) {
                        // Array to hold the file data.
                        byte[] fileContentBytes = new byte[fileContentLength];
                        // Read from the input stream into the fileContentBytes
array.
                        dataInputStream.readFully(fileContentBytes, 0,
fileContentBytes.length);
                        // Panel to hold the picture and file name.
                        JPanel jpFileRow = new JPanel();
                        jpFileRow.setLayout(new BoxLayout(jpFileRow,
BoxLayout.X_AXIS));
                        // Set the file name.
                        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")) {
                            // Set the name to be the fileId so you can get the
correct file from the panel.
                            jpFileRow.setName((String.valueOf(fileId)));
                            jpFileRow.addMouseListener(getMyMouseListener());
                            // Add everything.
                            jpFileRow.add(jlFileName);
                            jPanel.add(jpFileRow);
                            jFrame.validate();
                        } else {
                            // Set the name to be the fileId so you can get the
correct file from the panel.
                            jpFileRow.setName((String.valueOf(fileId)));
                            // Add a mouse listener so when it is clicked the
popup appears.
                            jpFileRow.addMouseListener(getMyMouseListener());
                            // Add the file name and pic type to the panel and
then add panel to parent panel.
                            jpFileRow.add(jlFileName);
                            jPanel.add(jpFileRow);
                            // Perform a relayout.
                            jFrame.validate();
                        }

                        // Add the new file to the array list which holds all our
data.
```

```java
                        myFiles.add(new MyFile(fileId, fileName,
fileContentBytes, getFileExtension(fileName)));
                        // Increment the fileId for the next file to be received.
                        fileId++;
                    }
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
        }
    }

    /**
     * @param fileName
     * @return The extension type of the file.
     */
    public static String getFileExtension(String fileName) {
        // Get the file type by using the last occurence of . (for example
aboutMe.txt returns txt).
        // Will have issues with files like myFile.tar.gz.
        int i = fileName.lastIndexOf('.');
        // If there is an extension.
        if (i > 0) {
            // Set the extension to the extension of the filename.
            return fileName.substring(i + 1);
        } else {
            return "No extension found.";
        }
    }

    /**
     * When the jpanel is clicked a popup shows to say whether the user wants to
download
     * the selected document.
     *
     * @return A mouselistener that is used by the jpanel.
     */
    public static MouseListener getMyMouseListener() {
        return new MouseListener() {
            @Override
            public void mouseClicked(MouseEvent e) {
                // Get the source of the click which is the JPanel.
                JPanel jPanel = (JPanel) e.getSource();
                // Get the ID of the file.
                int fileId = Integer.parseInt(jPanel.getName());
                // Loop through the file storage and see which file is the
selected one.
                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) {

        // Frame to hold everything.
        JFrame jFrame = new JFrame("Java File Downloader");
        // Set the size of the frame.
        jFrame.setSize(400, 400);

        // Panel to hold everything.
        JPanel jPanel = new JPanel();
        // Make the layout a box layout with child elements stacked on top of
each other.
```

```
        jPanel.setLayout(new BoxLayout(jPanel, BoxLayout.Y_AXIS));

        // Title above panel.
        JLabel jlTitle = new JLabel("Java File Downloader");
        // Center the label title horizontally.
        jlTitle.setAlignmentX(Component.CENTER_ALIGNMENT);
        // Change the font family, size, and style.
        jlTitle.setFont(new Font("Arial", Font.BOLD, 25));
        // Add spacing on the top and bottom of the element.
        jlTitle.setBorder(new EmptyBorder(20,0,10,0));

        // Label to prompt the user if they are sure they want to download the
file.
        JLabel jlPrompt = new JLabel("Are you sure you want to download " +
fileName + "?");
        // Change the font style, size, and family of the label.
        jlPrompt.setFont(new Font("Arial", Font.BOLD, 20));
        // Add spacing on the top and bottom of the label.
        jlPrompt.setBorder(new EmptyBorder(20,0,10,0));
        // Center the label horizontally.
        jlPrompt.setAlignmentX(Component.CENTER_ALIGNMENT);

        // Create the yes for accepting the download.
        JButton jbYes = new JButton("Yes");
        jbYes.setPreferredSize(new Dimension(150, 75));
        // Set the font for the button.
        jbYes.setFont(new Font("Arial", Font.BOLD, 20));

        // No button for rejecting the download.
        JButton jbNo = new JButton("No");
        // Change the size of the button must be preferred because if not the
layout will ignore it.
        jbNo.setPreferredSize(new Dimension(150, 75));
        // Set the font for the button.
        jbNo.setFont(new Font("Arial", Font.BOLD, 20));

        // Label to hold the content of the file whether it be text of images.
        JLabel jlFileContent = new JLabel();
        // Align the label horizontally.
        jlFileContent.setAlignmentX(Component.CENTER_ALIGNMENT);

        // Panel to hold the yes and no buttons and make the next to each other
left and right.
        JPanel jpButtons = new JPanel();
        // Add spacing around the panel.
        jpButtons.setBorder(new EmptyBorder(20, 0, 10, 0));
        // Add the yes and no buttons.
        jpButtons.add(jbYes);
        jpButtons.add(jbNo);

        // If the file is a text file then display the text.
        if (fileExtension.equalsIgnoreCase("txt")) {
            // Wrap it with <html> so that new lines are made.
            jlFileContent.setText("<html>" + new String(fileData) + "</html>");
            // If the file is not a text file then make it an image.
        } else {
            jlFileContent.setIcon(new ImageIcon(fileData));
        }

        // Yes so download file.
        jbYes.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // Create the file with its name.
                File fileToDownload = new File(fileName);
                try {
                    // Create a stream to write data to the file.
                    FileOutputStream fileOutputStream = new
FileOutputStream(fileToDownload);
                    // Write the actual file data to the file.
                    fileOutputStream.write(fileData);
                    // Close the stream.
                    fileOutputStream.close();
                    // Get rid of the jFrame. after the user clicked yes.
                    jFrame.dispose();
                } catch (IOException ex) {
                    ex.printStackTrace();
                }

            }
        });

        // No so close window.
        jbNo.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                // User clicked no so don't download the file but close the
jframe.
                jFrame.dispose();
            }
```

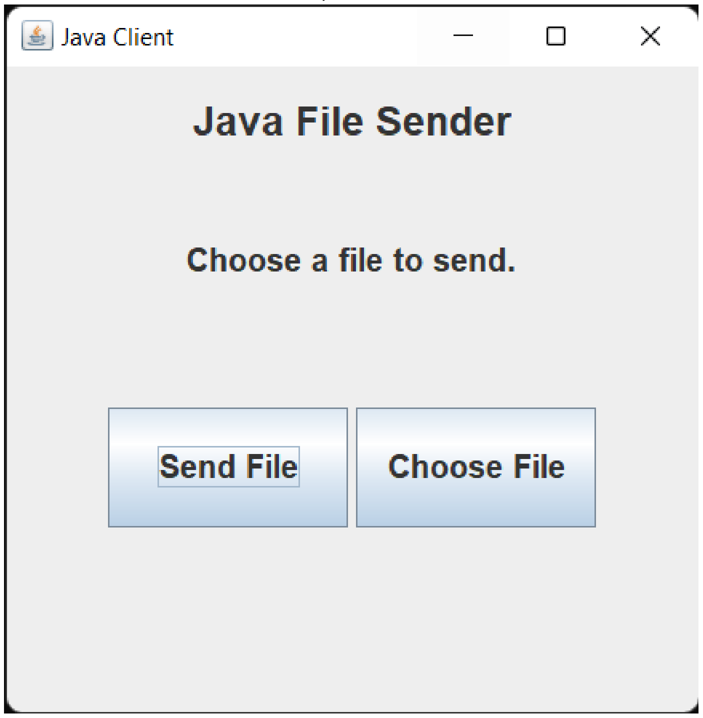
```java
        });

        // Add everything to the panel before adding to the frame.
        jPanel.add(jlTitle);
        jPanel.add(jlPrompt);
        jPanel.add(jlFileContent);
        jPanel.add(jpButtons);

        // Add panel to the frame.
        jFrame.add(jPanel);

        // Return the jFrame so it can be passed the right data and then shown.
        return jFrame;

    }


}
```

MyFile.java:

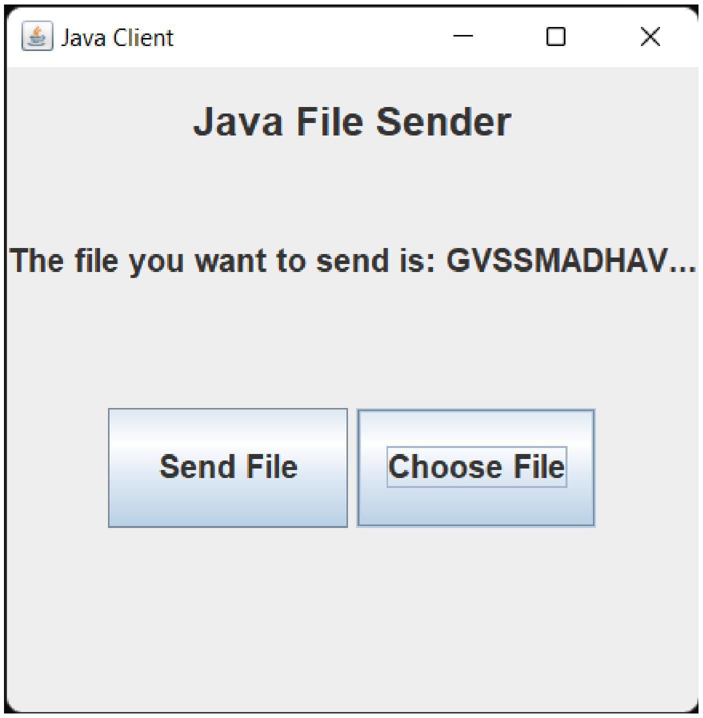
```java
/**
 * id - the id of the object
 * name - the name of the file
 * data - the file data
 * fileExtension - the file extension
 *
 */

public class MyFile {

    private int id;
    private String name;
    private byte[] data;
    private String fileExtension;

    public MyFile(int id, String name, byte[] data, String fileExtension) {
        this.id = id;
        this.name = name;
        this.data = data;
        this.fileExtension = fileExtension;
    }
    public void setId(int id) {
        this.id = id;
    }

    public void setName(String name) {
        this.name = name;
    }

    public void setData(byte[] data) {
        this.data = data;
    }

    public void setFileExtension(String fileExtension) {
        this.fileExtension = fileExtension;
    }

    public int getId() {
        return id;
    }

    public String getName() {
        return name;
    }

    public byte[] getData() {
        return data;
    }

    public String getFileExtension() {
        return fileExtension;
    }
}
```

Output:

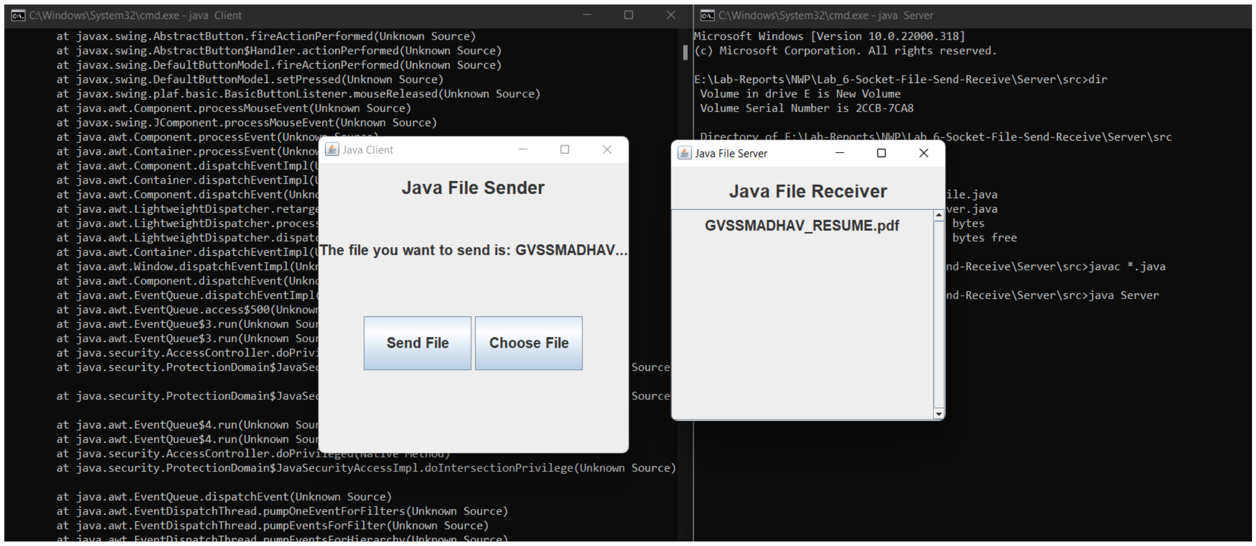
Initial Screen of Client Side, When We Start The Client



The Screen After We Click Choose File And Select A File



Now We Need To Start The Server before Clicking Send File, And After The Server Pops Up, when we click Send File, The File Will be Visible in the Menu of Receiver Box as below



C:\Windows\System32\cmd.exe - java Client

```
        at javax.swing.AbstractButton.fireActionPerformed(Unknown Source)
        at javax.swing.AbstractButton$Handler.actionPerformed(Unknown Source)
        at javax.swing.DefaultButtonModel.fireActionPerformed(Unknown Source)
        at javax.swing.DefaultButtonModel.setPressed(Unknown Source)
        at javax.swing.plaf.basic.BasicButtonListener.mouseReleased(Unknown Source)
        at java.awt.Component.processMouseEvent(Unknown Source)
        at javax.swing.JComponent.processMouseEvent(Unknown Source)
        at java.awt.Component.processEvent(Unknow
        at java.awt.Container.processEvent(Unknow
        at java.awt.Component.dispatchEventImpl(U
        at java.awt.Container.dispatchEventImpl(U
        at java.awt.Component.dispatchEvent(Unkno
        at java.awt.LightweightDispatcher.retarge
        at java.awt.LightweightDispatcher.process
        at java.awt.LightweightDispatcher.dispatc
        at java.awt.Container.dispatchEventImpl(U
        at java.awt.Window.dispatchEventImpl(Unkr
        at java.awt.Component.dispatchEvent(Unkno
        at java.awt.EventQueue.dispatchEventImpl(
        at java.awt.EventQueue.access$500(Unknown
        at java.awt.EventQueue$3.run(Unknown Sour
        at java.awt.EventQueue$3.run(Unknown Sour
        at java.security.AccessController.doPrivi
        at java.security.ProtectionDomain$JavaSec                                 Source

        at java.security.ProtectionDomain$JavaSec                                 Source

        at java.awt.EventQueue$4.run(Unknown Sour
        at java.awt.EventQueue$4.run(Unknown Sour
        at java.security.AccessController.doPrivi
        at java.security.ProtectionDomain$JavaSecurityAccessImpl.doIntersectionPrivilege(Unknown Source)

        at java.awt.EventQueue.dispatchEvent(Unknown Source)
        at java.awt.EventDispatchThread.pumpOneEventForFilters(Unknown Source)
        at java.awt.EventDispatchThread.pumpEventsForFilter(Unknown Source)
```

Java Client

Java File Sender

The file you want to send is: GVSSMADHAV...

C:\Windows\System32\cmd.exe - java Server

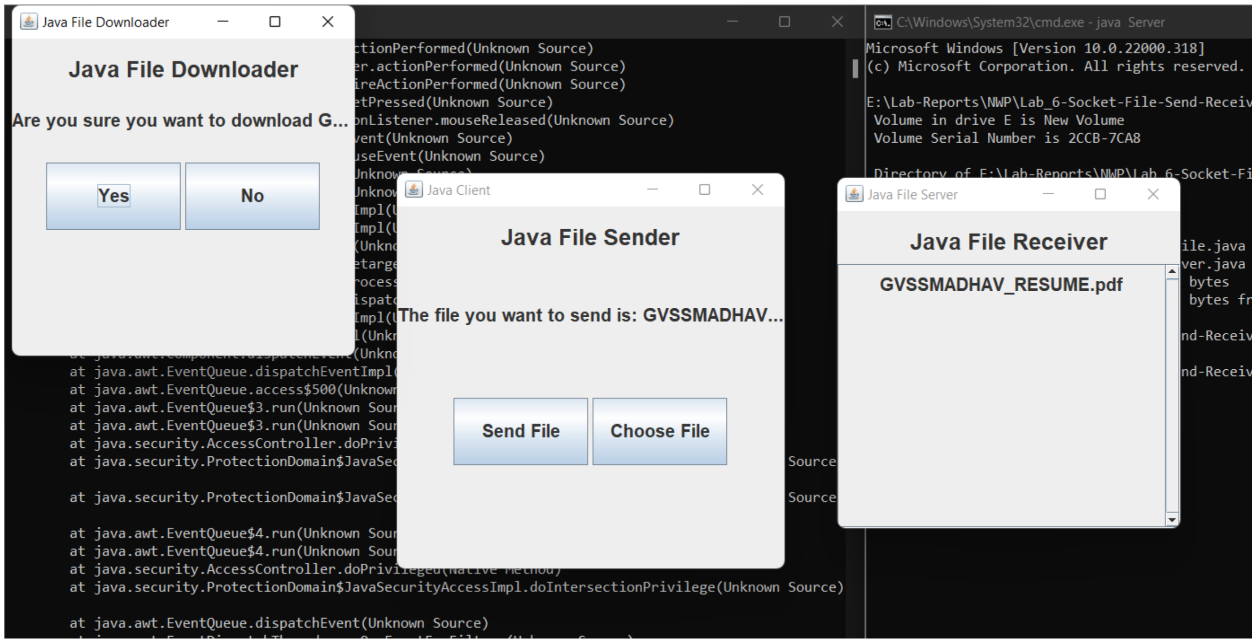
```
Microsoft Windows [Version 10.0.22000.318]
(c) Microsoft Corporation. All rights reserved.

E:\Lab-Reports\NWP\Lab_6-Socket-File-Send-Receive\Server\src>dir
 Volume in drive E is New Volume
 Volume Serial Number is 2CCB-7CA8

 Directory of E:\Lab-Reports\NWP\Lab_6-Socket-File-Send-Receive\Server\src

                                    ile.java
                                    ver.java
                                     bytes
                                     bytes free

                                    nd-Receive\Server\src>javac *.java

                                    nd-Receive\Server\src>java Server
```

Java File Server

Java File Receiver

GVSSMADHAV_RESUME.pdf

When we click on the file name in the server, a new pop up by downloader comes up and when we click on yes, it get's stored in the source folder of server and we can view it manually



Java File Downloader

Java File Downloader

Are you sure you want to download G...

Java Client

Java File Sender

The file you want to send is: GVSSMADHAV...

Java File Server

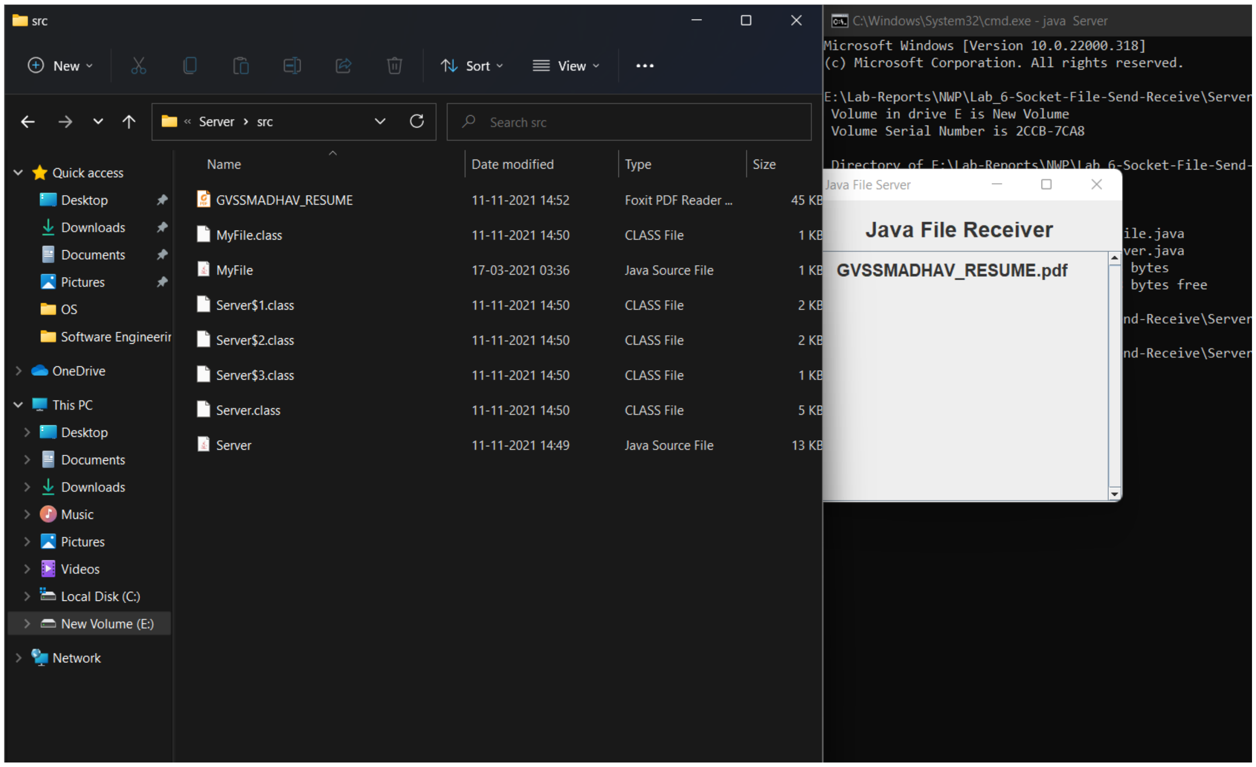
Java File Receiver

GVSSMADHAV_RESUME.pdf

C:\Windows\System32\cmd.exe - java Server

```
Microsoft Windows [Version 10.0.22000.318]
(c) Microsoft Corporation. All rights reserved.

E:\Lab-Reports\NWP\Lab_6-Socket-File-Send-Receiv
 Volume in drive E is New Volume
 Volume Serial Number is 2CCB-7CA8
```

Here, I've Sent My Resume From E Directory and After Clicking on Download, we can see it's visible in the source folder of the client.



Name	Date modified	Type	Size
GVSSMADHAV_RESUME	11-11-2021 14:52	Foxit PDF Reader ...	45 KB
MyFile.class	11-11-2021 14:50	CLASS File	1 KB
MyFile	17-03-2021 03:36	Java Source File	1 KB
Server$1.class	11-11-2021 14:50	CLASS File	2 KB
Server$2.class	11-11-2021 14:50	CLASS File	2 KB
Server$3.class	11-11-2021 14:50	CLASS File	1 KB
Server.class	11-11-2021 14:50	CLASS File	5 KB
Server	11-11-2021 14:49	Java Source File	13 KB

C:\Windows\System32\cmd.exe - java Server

```
Microsoft Windows [Version 10.0.22000.318]
(c) Microsoft Corporation. All rights reserved.

E:\Lab-Reports\NWP\Lab_6-Socket-File-Send-Receive\Server
 Volume in drive E is New Volume
 Volume Serial Number is 2CCB-7CA8

 Directory of E:\Lab-Reports\NWP\Lab_6-Socket-File-Send-

                                        ile.java
                                        ver.java
                                         bytes
                                         bytes free

                                        nd-Receive\Server

                                        nd-Receive\Server
```

Java File Server

Java File Receiver

GVSSMADHAV_RESUME.pdf