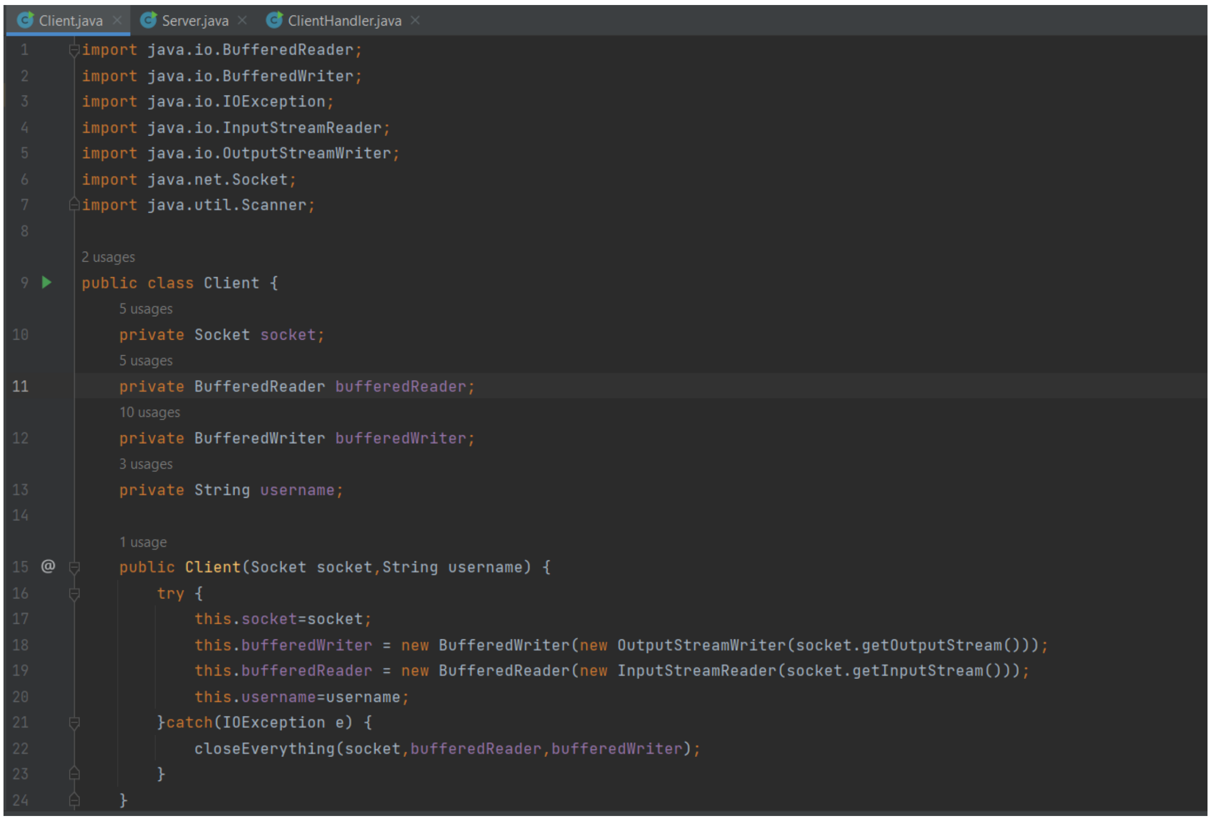
PRACTICAL NO. 4

Name: Saloni Vishwakarma(C1-13)

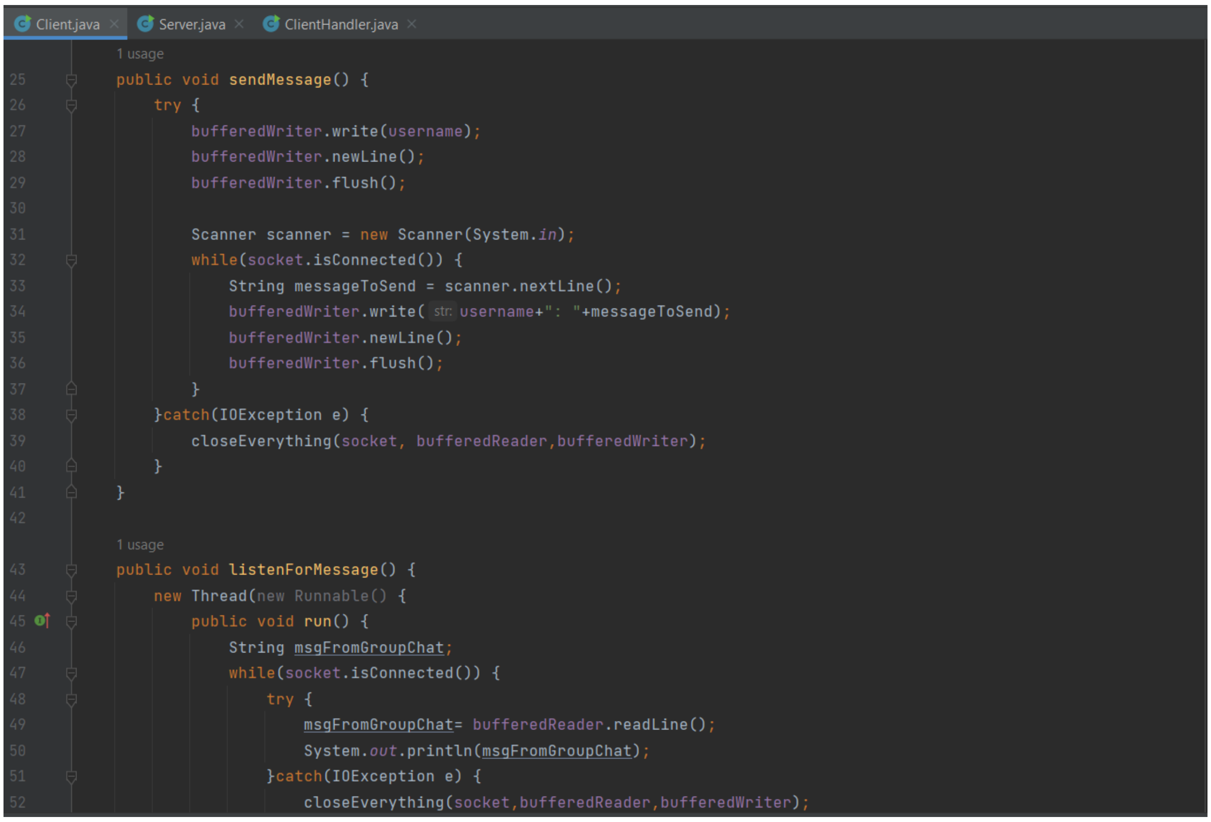
Aim: Write a program based on socket programming in JAVA for file transfer, image transfer and multiple-chat box.

a) Multiple-Chat Box (Codes):

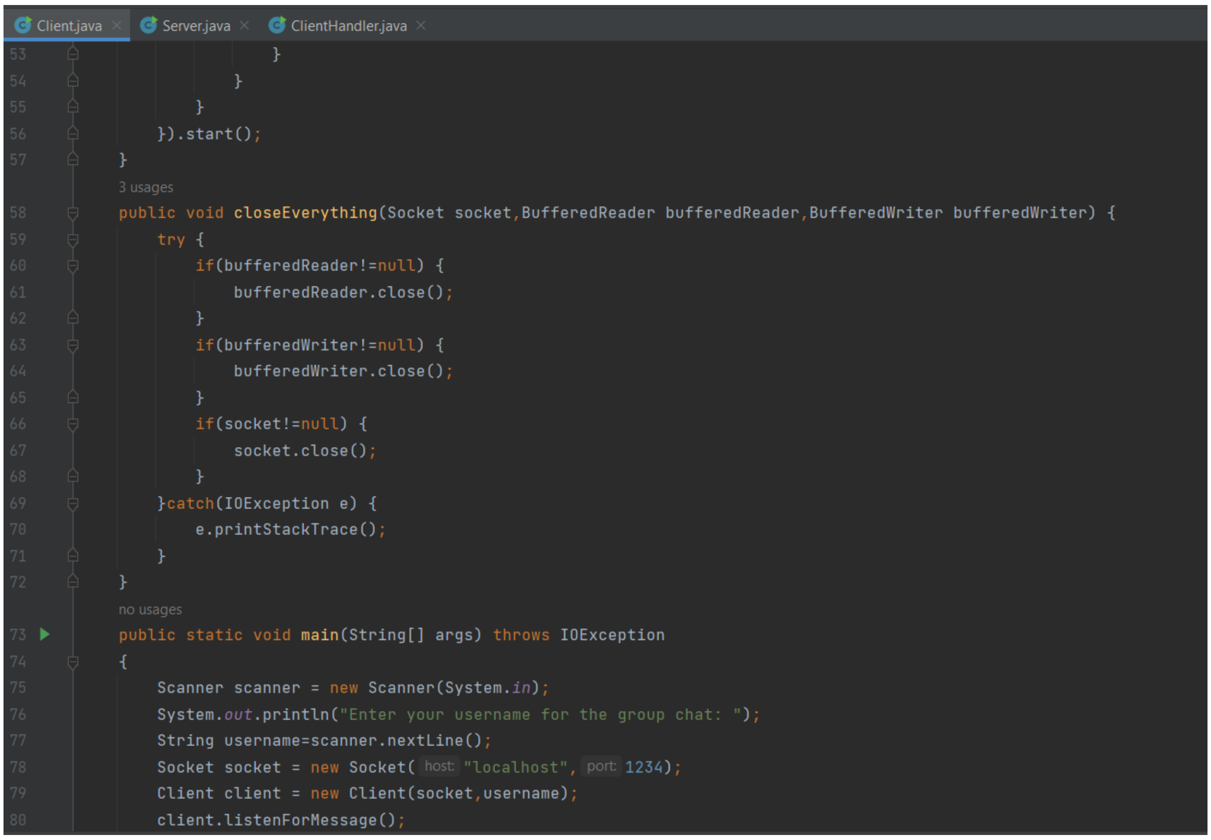
Client:


Client.java Server.java ClientHandler.java

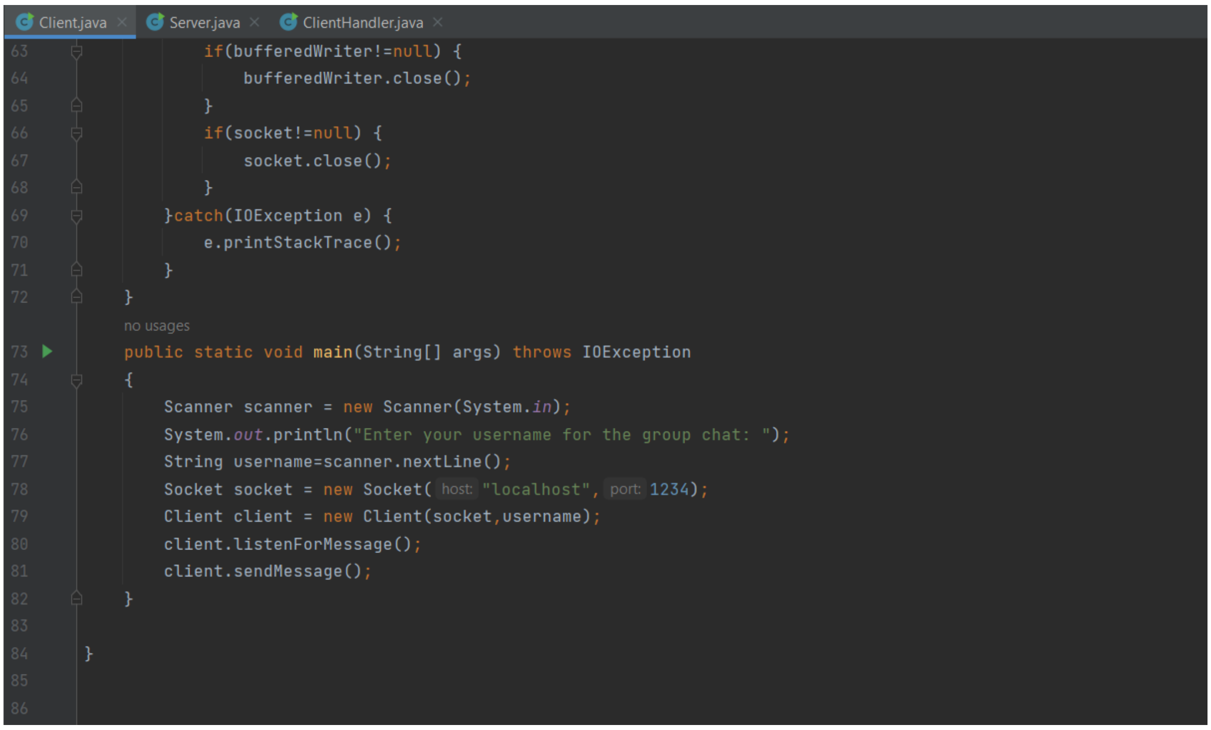
```
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.Socket;
import java.util.Scanner;

2 usages
public class Client {
    5 usages
    private Socket socket;
    5 usages
    private BufferedReader bufferedReader;
    10 usages
    private BufferedWriter bufferedWriter;
    3 usages
    private String username;

    1 usage
    public Client(Socket socket,String username) {
        try {
            this.socket=socket;
            this.bufferedWriter = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            this.bufferedReader = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            this.username=username;
        }catch(IOException e) {
            closeEverything(socket,bufferedReader,bufferedWriter);
        }
    }
```



```java
    1 usage
    public void sendMessage() {
        try {
            bufferedWriter.write(username);
            bufferedWriter.newLine();
            bufferedWriter.flush();

            Scanner scanner = new Scanner(System.in);
            while(socket.isConnected()) {
                String messageToSend = scanner.nextLine();
                bufferedWriter.write( str: username+": "+messageToSend);
                bufferedWriter.newLine();
                bufferedWriter.flush();
            }
        }catch(IOException e) {
            closeEverything(socket, bufferedReader,bufferedWriter);
        }
    }

    1 usage
    public void listenForMessage() {
        new Thread(new Runnable() {
            public void run() {
                String msgFromGroupChat;
                while(socket.isConnected()) {
                    try {
                        msgFromGroupChat= bufferedReader.readLine();
                        System.out.println(msgFromGroupChat);
                    }catch(IOException e) {
                        closeEverything(socket,bufferedReader,bufferedWriter);
```

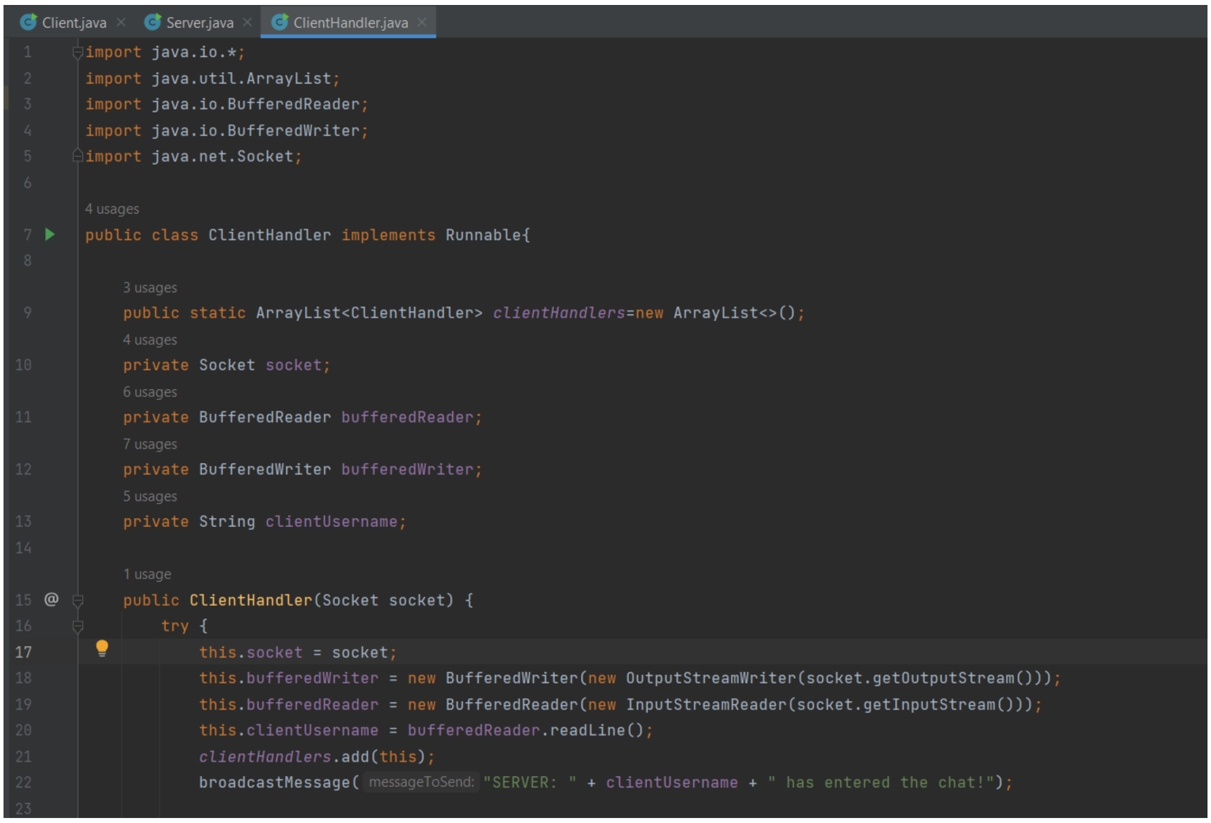


```java
                    }
                }
            }
        }).start();
    }
    3 usages
    public void closeEverything(Socket socket,BufferedReader bufferedReader,BufferedWriter bufferedWriter) {
        try {
            if(bufferedReader!=null) {
                bufferedReader.close();
            }
            if(bufferedWriter!=null) {
                bufferedWriter.close();
            }
            if(socket!=null) {
                socket.close();
            }
        }catch(IOException e) {
            e.printStackTrace();
        }
    }
    no usages
    public static void main(String[] args) throws IOException
    {
        Scanner scanner = new Scanner(System.in);
        System.out.println("Enter your username for the group chat: ");
        String username=scanner.nextLine();
        Socket socket = new Socket( host: "localhost", port: 1234);
        Client client = new Client(socket,username);
        client.listenForMessage();
```

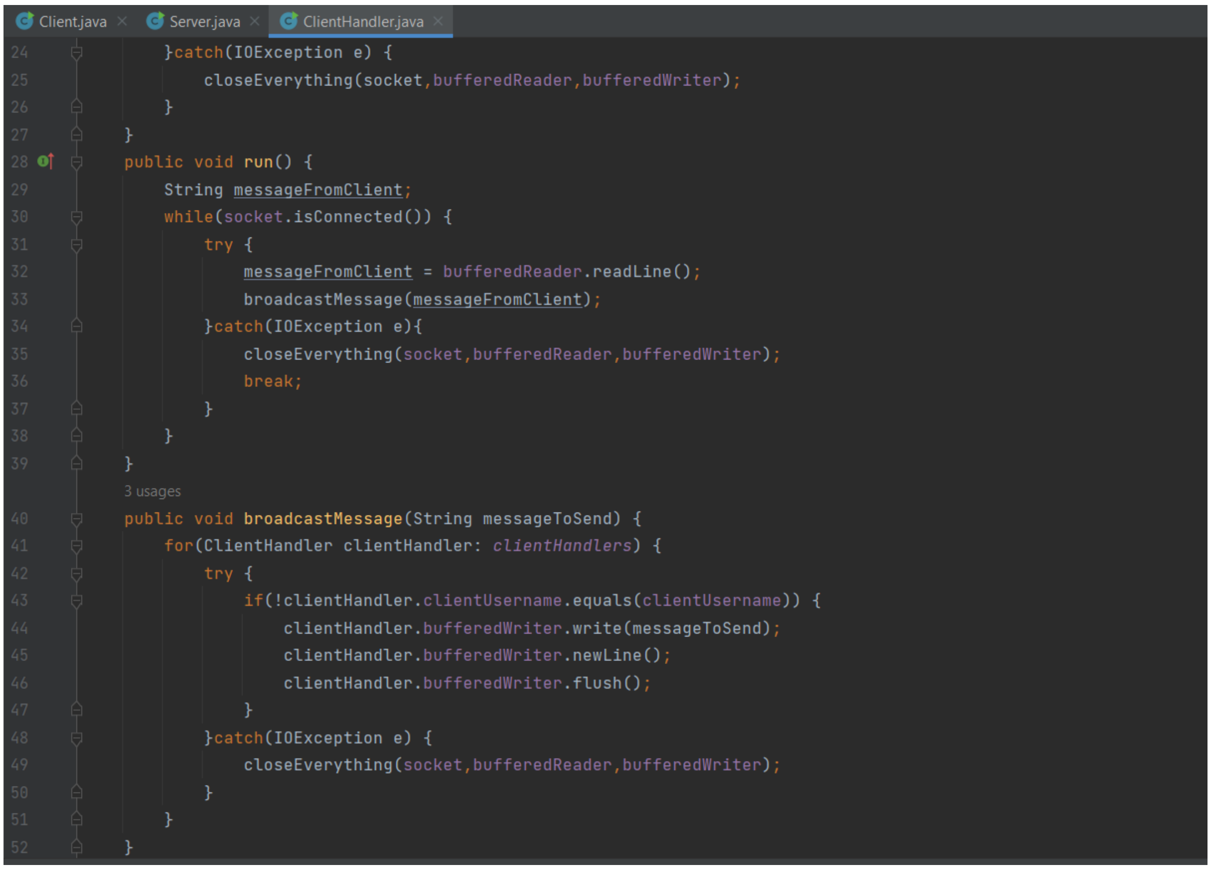


```java
                if(bufferedWriter!=null) {
                    bufferedWriter.close();
                }
                if(socket!=null) {
                    socket.close();
                }
            }catch(IOException e) {
                e.printStackTrace();
            }
        }
        no usages
        public static void main(String[] args) throws IOException
        {
            Scanner scanner = new Scanner(System.in);
            System.out.println("Enter your username for the group chat: ");
            String username=scanner.nextLine();
            Socket socket = new Socket( host: "localhost", port: 1234);
            Client client = new Client(socket,username);
            client.listenForMessage();
            client.sendMessage();
        }

    }
```

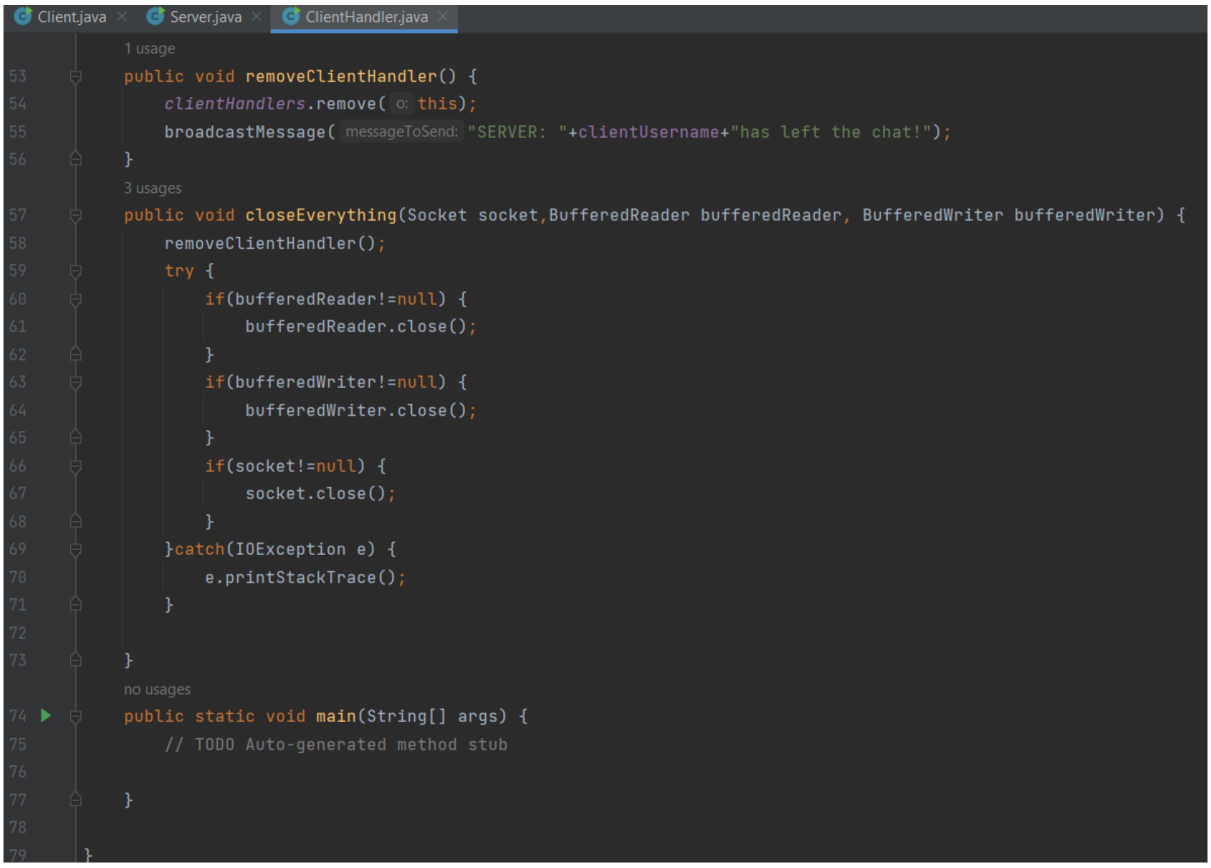
ClientHandler:



```java
import java.io.*;
import java.util.ArrayList;
import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.net.Socket;

4 usages
public class ClientHandler implements Runnable{

    3 usages
    public static ArrayList<ClientHandler> clientHandlers=new ArrayList<>();
    4 usages
    private Socket socket;
    6 usages
    private BufferedReader bufferedReader;
    7 usages
    private BufferedWriter bufferedWriter;
    5 usages
    private String clientUsername;

    1 usage
    public ClientHandler(Socket socket) {
        try {
            this.socket = socket;
            this.bufferedWriter = new BufferedWriter(new OutputStreamWriter(socket.getOutputStream()));
            this.bufferedReader = new BufferedReader(new InputStreamReader(socket.getInputStream()));
            this.clientUsername = bufferedReader.readLine();
            clientHandlers.add(this);
            broadcastMessage( messageToSend: "SERVER: " + clientUsername + " has entered the chat!");
```

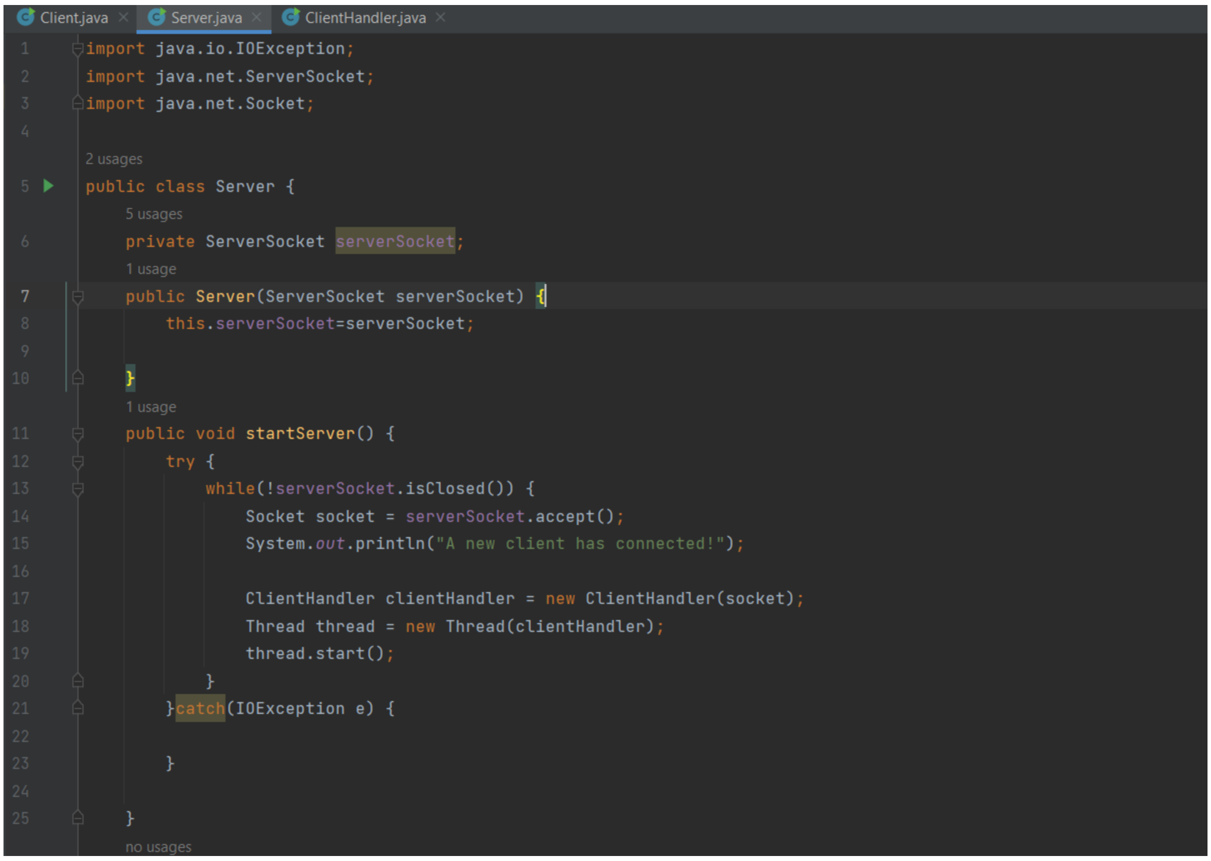


```java
        }catch(IOException e) {
            closeEverything(socket,bufferedReader,bufferedWriter);
        }
    }
    public void run() {
        String messageFromClient;
        while(socket.isConnected()) {
            try {
                messageFromClient = bufferedReader.readLine();
                broadcastMessage(messageFromClient);
            }catch(IOException e){
                closeEverything(socket,bufferedReader,bufferedWriter);
                break;
            }
        }
    }
    3 usages
    public void broadcastMessage(String messageToSend) {
        for(ClientHandler clientHandler: clientHandlers) {
            try {
                if(!clientHandler.clientUsername.equals(clientUsername)) {
                    clientHandler.bufferedWriter.write(messageToSend);
                    clientHandler.bufferedWriter.newLine();
                    clientHandler.bufferedWriter.flush();
                }
            }catch(IOException e) {
                closeEverything(socket,bufferedReader,bufferedWriter);
            }
        }
    }
```



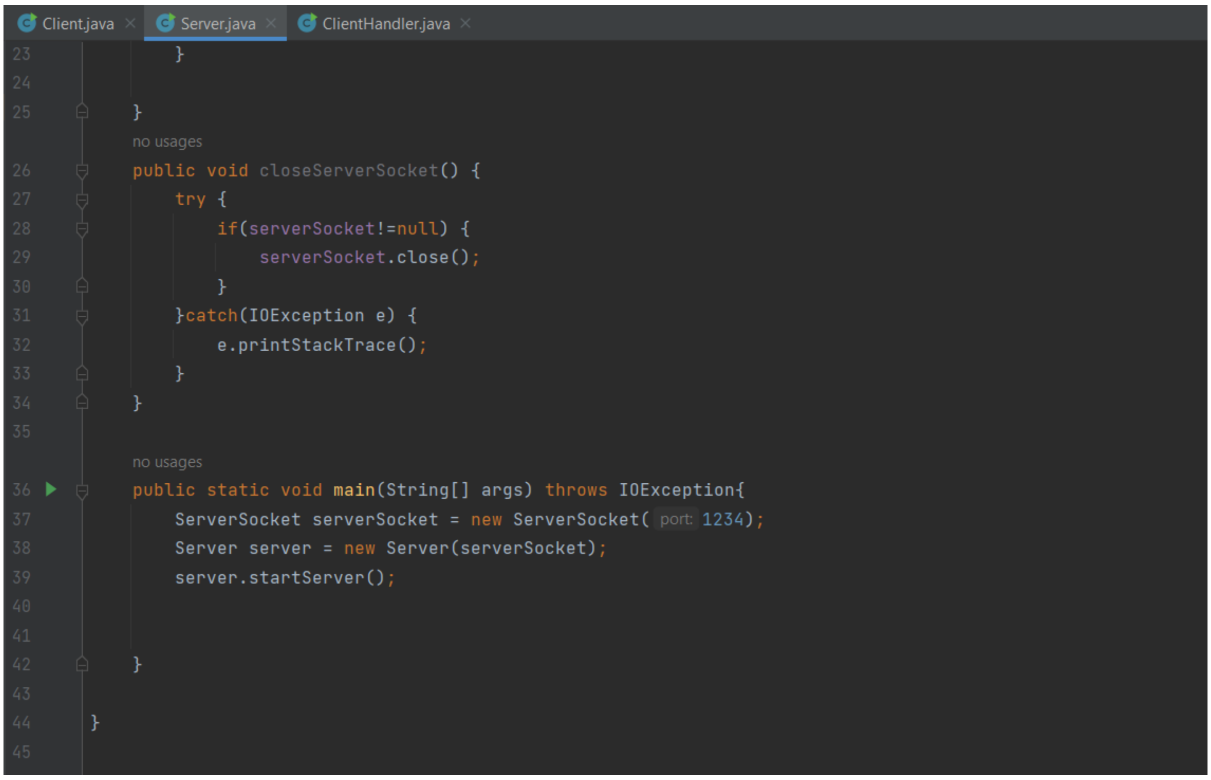
```java
    1 usage
    public void removeClientHandler() {
        clientHandlers.remove( o: this);
        broadcastMessage( messageToSend: "SERVER: "+clientUsername+"has left the chat!");
    }
    3 usages
    public void closeEverything(Socket socket,BufferedReader bufferedReader, BufferedWriter bufferedWriter) {
        removeClientHandler();
        try {
            if(bufferedReader!=null) {
                bufferedReader.close();
            }
            if(bufferedWriter!=null) {
                bufferedWriter.close();
            }
            if(socket!=null) {
                socket.close();
            }
        }catch(IOException e) {
            e.printStackTrace();
        }

    }
    no usages
    public static void main(String[] args) {
        // TODO Auto-generated method stub

    }

}
```

Server:



Client.java   Server.java   ClientHandler.java

```
import java.io.IOException;
import java.net.ServerSocket;
import java.net.Socket;

2 usages
public class Server {
    5 usages
    private ServerSocket serverSocket;
    1 usage
    public Server(ServerSocket serverSocket) {
        this.serverSocket=serverSocket;

    }
    1 usage
    public void startServer() {
        try {
            while(!serverSocket.isClosed()) {
                Socket socket = serverSocket.accept();
                System.out.println("A new client has connected!");

                ClientHandler clientHandler = new ClientHandler(socket);
                Thread thread = new Thread(clientHandler);
                thread.start();
            }
        }catch(IOException e) {

        }

    }
    no usages
```

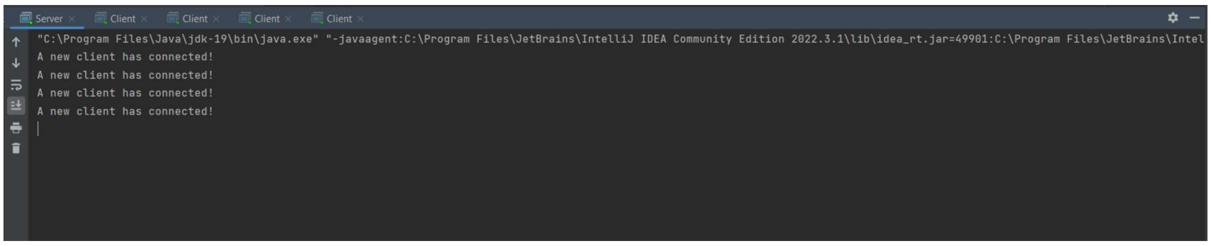


Client.java   Server.java   ClientHandler.java

```
        }

    }
    no usages
    public void closeServerSocket() {
        try {
            if(serverSocket!=null) {
                serverSocket.close();
            }
        }catch(IOException e) {
            e.printStackTrace();
        }
    }

    no usages
    public static void main(String[] args) throws IOException{
        ServerSocket serverSocket = new ServerSocket( port: 1234);
        Server server = new Server(serverSocket);
        server.startServer();


    }

}

```

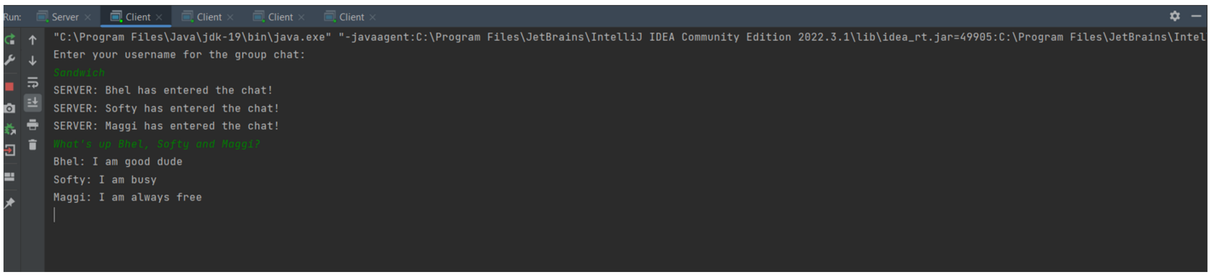
Multiple-Chat Box (Outputs):

Server



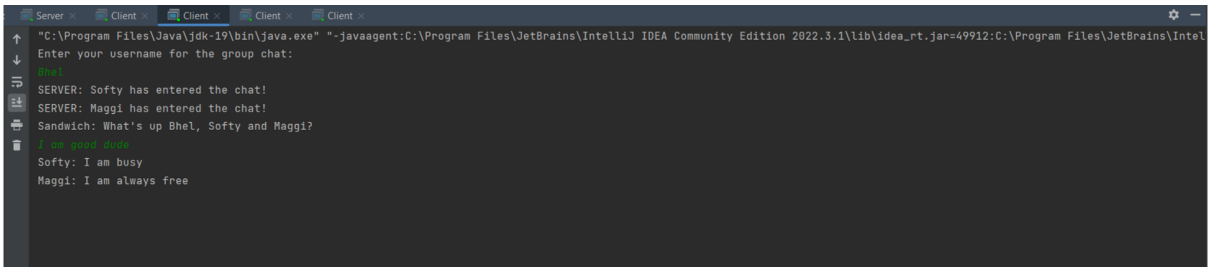
```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=49901:C:\Program Files\JetBrains\Intel
A new client has connected!
A new client has connected!
A new client has connected!
A new client has connected!
```

Client-1



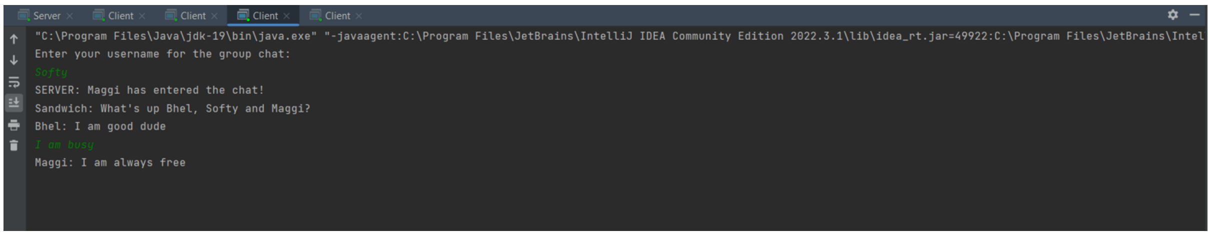
```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=49905:C:\Program Files\JetBrains\Intel
Enter your username for the group chat:
Sandwich
SERVER: Bhel has entered the chat!
SERVER: Softy has entered the chat!
SERVER: Maggi has entered the chat!
What's up Bhel, Softy and Maggi?
Bhel: I am good dude
Softy: I am busy
Maggi: I am always free
```

Client-2



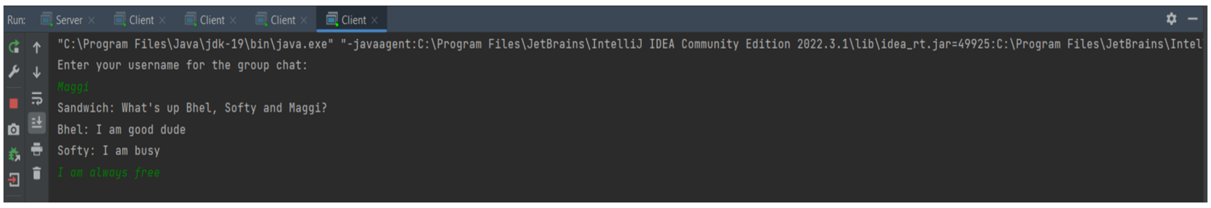
```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=49912:C:\Program Files\JetBrains\Intel
Enter your username for the group chat:
Bhel
SERVER: Softy has entered the chat!
SERVER: Maggi has entered the chat!
Sandwich: What's up Bhel, Softy and Maggi?
I am good dude
Softy: I am busy
Maggi: I am always free
```

Client-3



```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=49922:C:\Program Files\JetBrains\Intel
Enter your username for the group chat:
Softy
SERVER: Maggi has entered the chat!
Sandwich: What's up Bhel, Softy and Maggi?
Bhel: I am good dude
I am busy
Maggi: I am always free
```

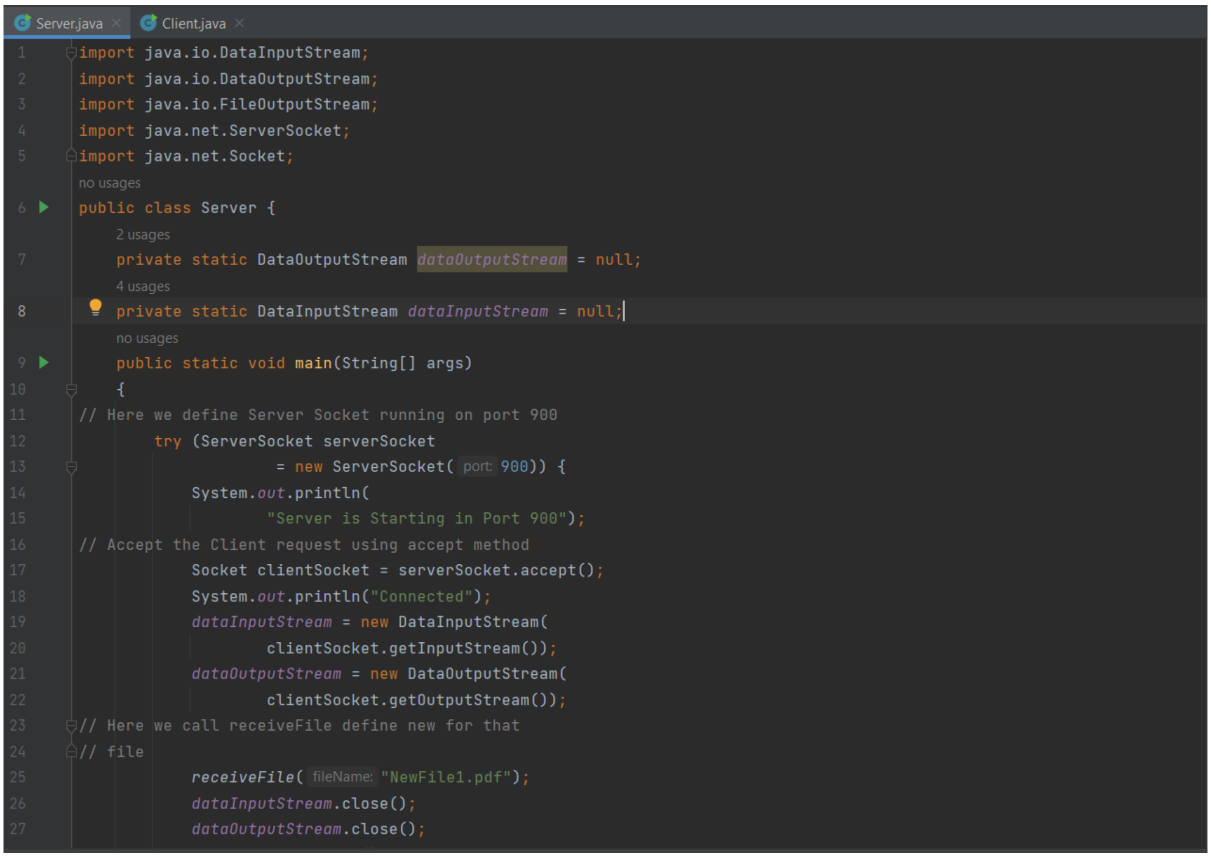
Client-4



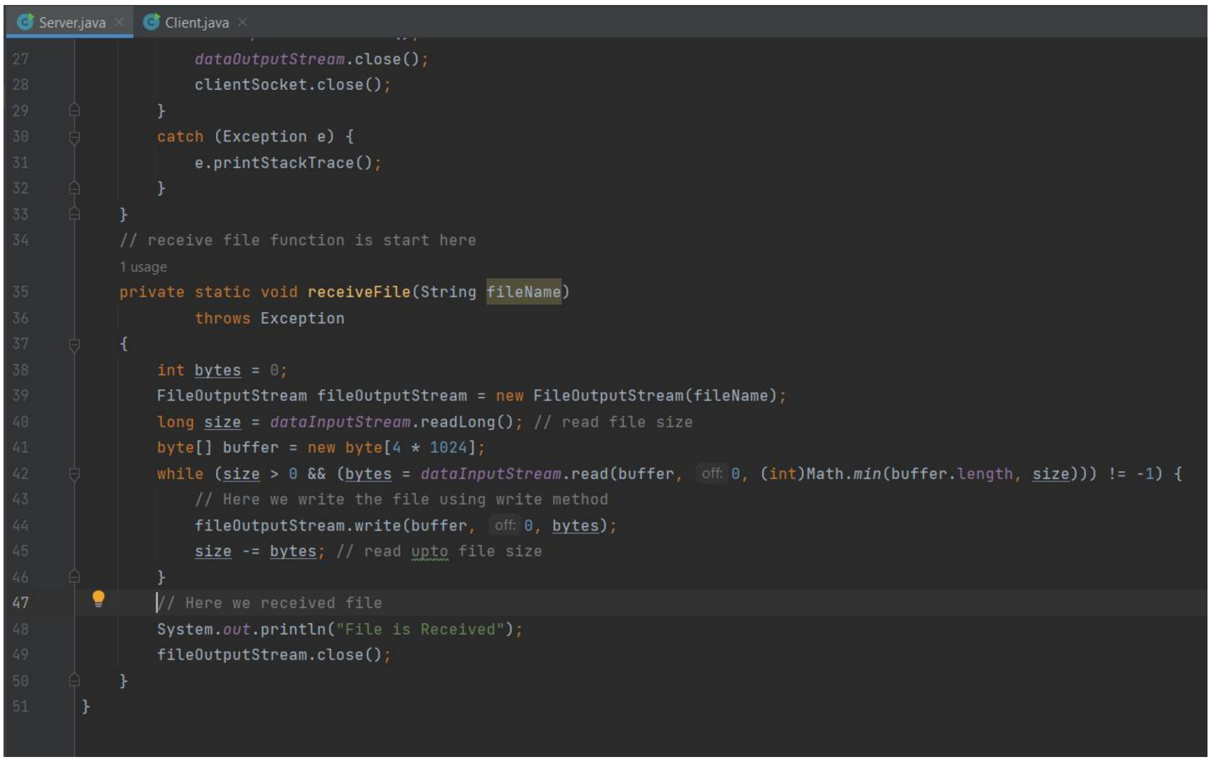
```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=49925:C:\Program Files\JetBrains\Intel
Enter your username for the group chat:
Maggi
Sandwich: What's up Bhel, Softy and Maggi?
Bhel: I am good dude
Softy: I am busy
I am always free
```

b) File Transfer (Codes):

Server:

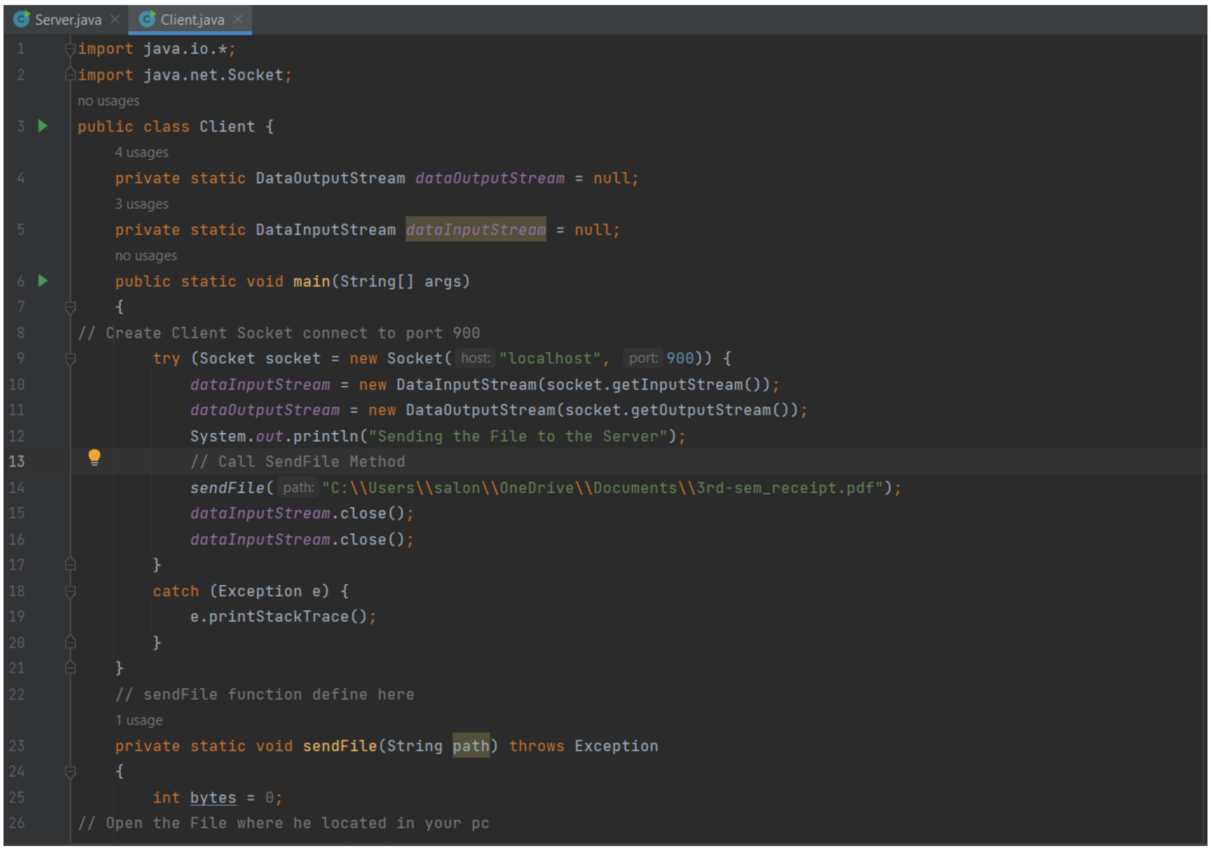


```java
import java.io.DataInputStream;
import java.io.DataOutputStream;
import java.io.FileOutputStream;
import java.net.ServerSocket;
import java.net.Socket;
no usages
public class Server {
    2 usages
    private static DataOutputStream dataOutputStream = null;
    4 usages
    private static DataInputStream dataInputStream = null;
    no usages
    public static void main(String[] args)
    {
// Here we define Server Socket running on port 900
        try (ServerSocket serverSocket
                     = new ServerSocket( port: 900)) {
            System.out.println(
                    "Server is Starting in Port 900");
// Accept the Client request using accept method
            Socket clientSocket = serverSocket.accept();
            System.out.println("Connected");
            dataInputStream = new DataInputStream(
                    clientSocket.getInputStream());
            dataOutputStream = new DataOutputStream(
                    clientSocket.getOutputStream());
// Here we call receiveFile define new for that
// file
            receiveFile( fileName: "NewFile1.pdf");
            dataInputStream.close();
            dataOutputStream.close();
```



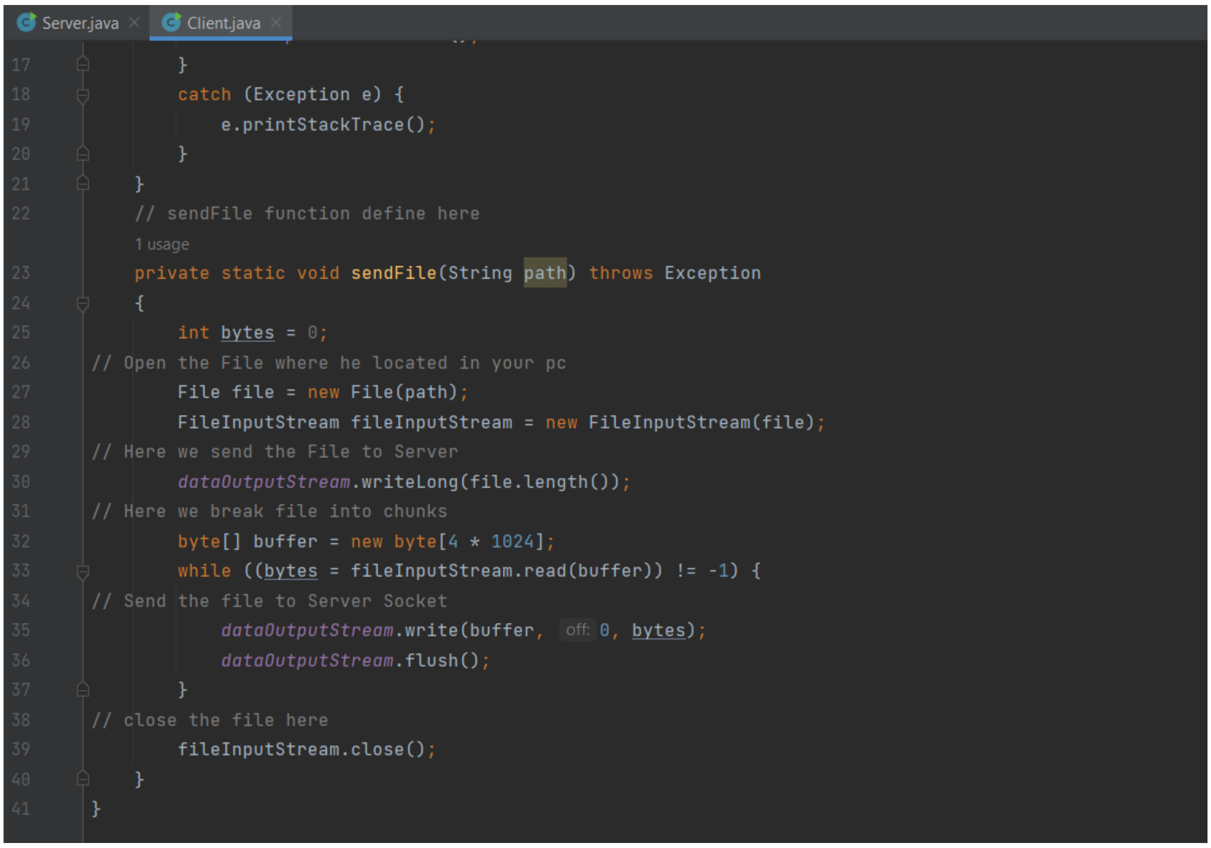
```java
            dataOutputStream.close();
            clientSocket.close();
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
    // receive file function is start here
    1 usage
    private static void receiveFile(String fileName)
            throws Exception
    {
        int bytes = 0;
        FileOutputStream fileOutputStream = new FileOutputStream(fileName);
        long size = dataInputStream.readLong(); // read file size
        byte[] buffer = new byte[4 * 1024];
        while (size > 0 && (bytes = dataInputStream.read(buffer,  off: 0, (int)Math.min(buffer.length, size))) != -1) {
            // Here we write the file using write method
            fileOutputStream.write(buffer,  off: 0, bytes);
            size -= bytes; // read upto file size
        }
        // Here we received file
        System.out.println("File is Received");
        fileOutputStream.close();
    }
}
```

Client:



Server.java Client.java

```
import java.io.*;
import java.net.Socket;
no usages
public class Client {
    4 usages
    private static DataOutputStream dataOutputStream = null;
    3 usages
    private static DataInputStream dataInputStream = null;
    no usages
    public static void main(String[] args)
    {
// Create Client Socket connect to port 900
        try (Socket socket = new Socket( host: "localhost",  port: 900)) {
            dataInputStream = new DataInputStream(socket.getInputStream());
            dataOutputStream = new DataOutputStream(socket.getOutputStream());
            System.out.println("Sending the File to the Server");
            // Call SendFile Method
            sendFile( path: "C:\\Users\\salon\\OneDrive\\Documents\\3rd-sem_receipt.pdf");
            dataInputStream.close();
            dataInputStream.close();
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
    // sendFile function define here
    1 usage
    private static void sendFile(String path) throws Exception
    {
        int bytes = 0;
// Open the File where he located in your pc
```

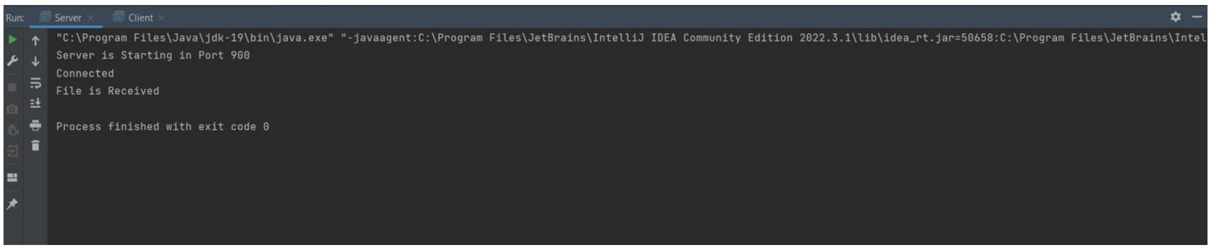


Server.java Client.java

```
        }
        catch (Exception e) {
            e.printStackTrace();
        }
    }
    // sendFile function define here
    1 usage
    private static void sendFile(String path) throws Exception
    {
        int bytes = 0;
// Open the File where he located in your pc
        File file = new File(path);
        FileInputStream fileInputStream = new FileInputStream(file);
// Here we send the File to Server
        dataOutputStream.writeLong(file.length());
// Here we break file into chunks
        byte[] buffer = new byte[4 * 1024];
        while ((bytes = fileInputStream.read(buffer)) != -1) {
// Send the file to Server Socket
            dataOutputStream.write(buffer,  off: 0, bytes);
            dataOutputStream.flush();
        }
// close the file here
        fileInputStream.close();
    }
}
```

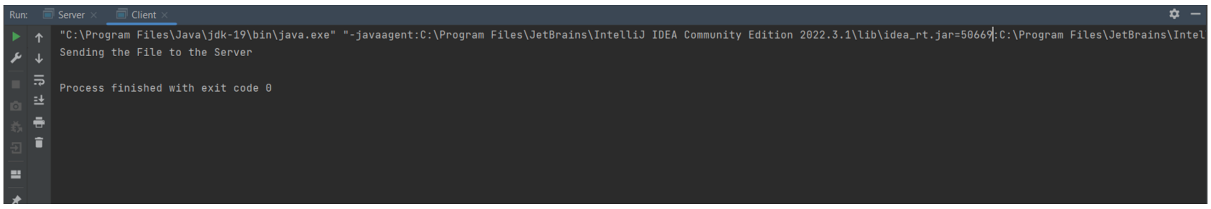
File Transfer (Outputs):

Server



```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=50658:C:\Program Files\JetBrains\Intel
Server is Starting in Port 900
Connected
File is Received

Process finished with exit code 0
```

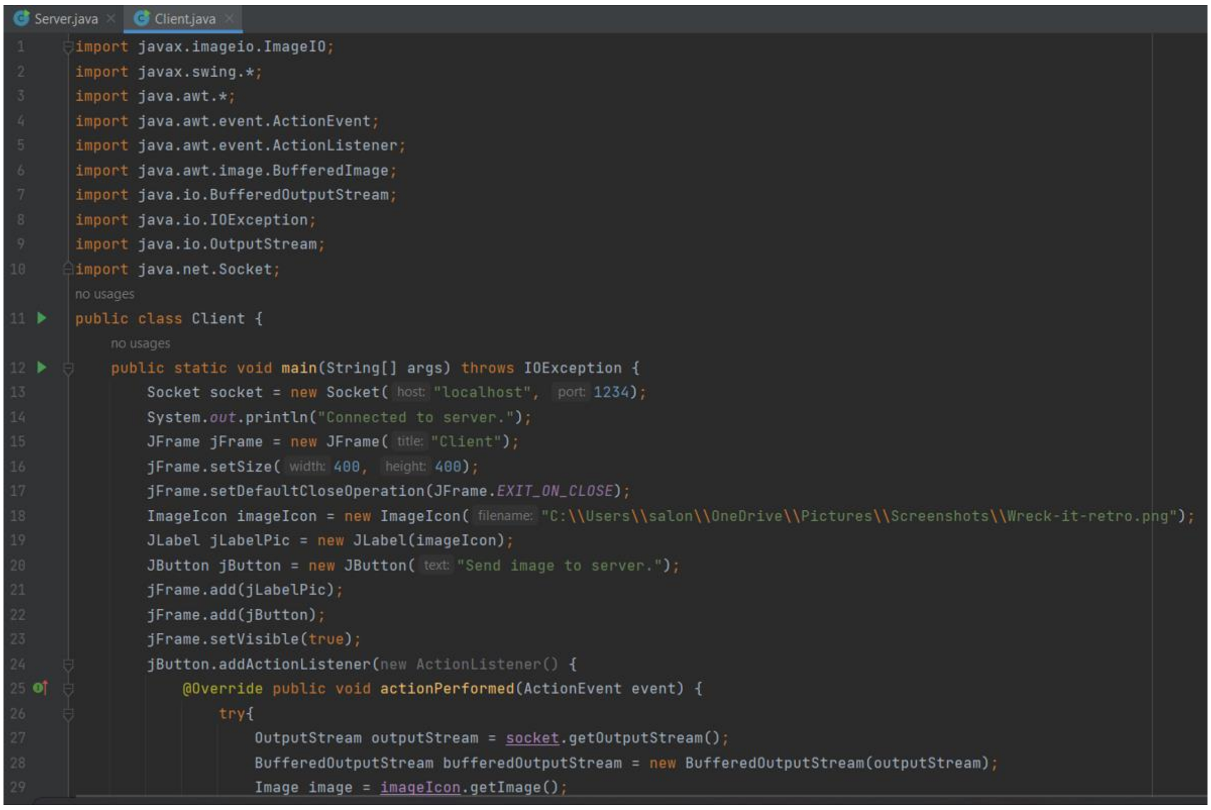
Client



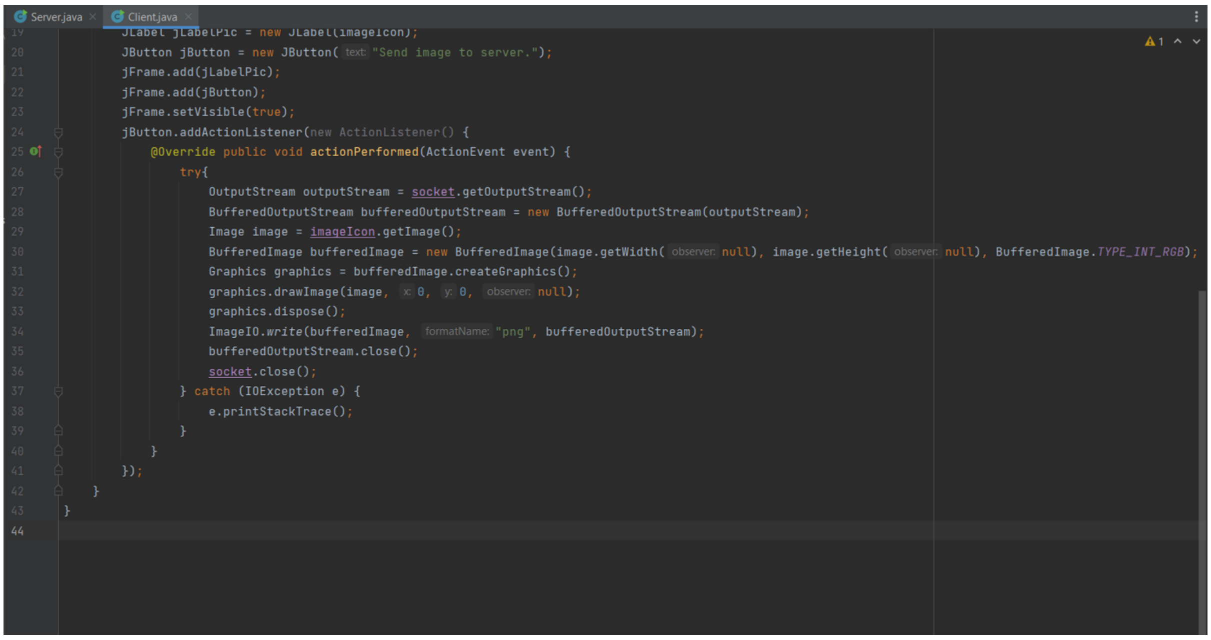
```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=50669:C:\Program Files\JetBrains\Intel
Sending the File to the Server

Process finished with exit code 0
```

c) Image-transfer (Codes):

Client:

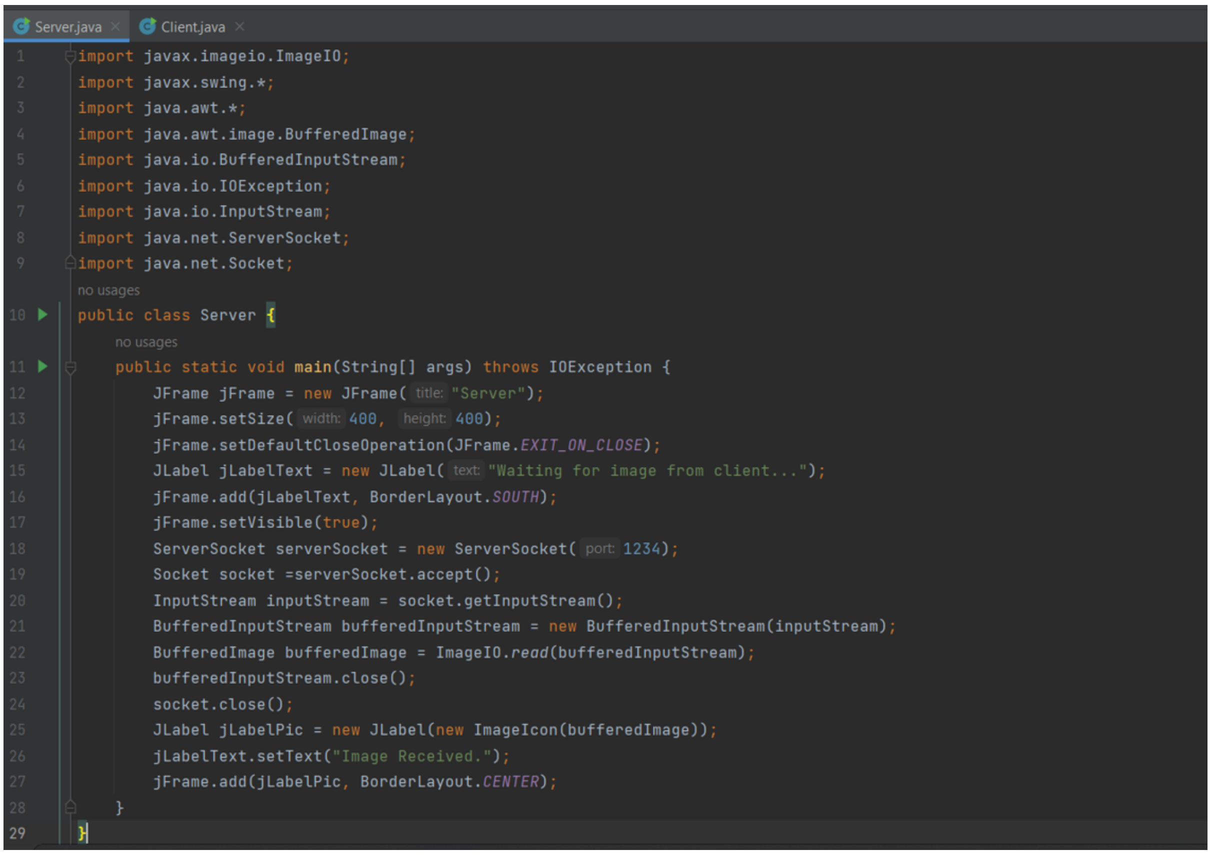


```java
import javax.imageio.ImageIO;
import javax.swing.*;
import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.image.BufferedImage;
import java.io.BufferedOutputStream;
import java.io.IOException;
import java.io.OutputStream;
import java.net.Socket;
no usages
public class Client {
    no usages
    public static void main(String[] args) throws IOException {
        Socket socket = new Socket( host: "localhost",  port: 1234);
        System.out.println("Connected to server.");
        JFrame jFrame = new JFrame( title: "Client");
        jFrame.setSize( width: 400,  height: 400);
        jFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        ImageIcon imageIcon = new ImageIcon( filename: "C:\\Users\\salon\\OneDrive\\Pictures\\Screenshots\\Wreck-it-retro.png");
        JLabel jLabelPic = new JLabel(imageIcon);
        JButton jButton = new JButton( text: "Send image to server.");
        jFrame.add(jLabelPic);
        jFrame.add(jButton);
        jFrame.setVisible(true);
        jButton.addActionListener(new ActionListener() {
            @Override public void actionPerformed(ActionEvent event) {
                try{
                    OutputStream outputStream = socket.getOutputStream();
                    BufferedOutputStream bufferedOutputStream = new BufferedOutputStream(outputStream);
                    Image image = imageIcon.getImage();
```


Server.java × Client.java ×

```
        JLabel jLabelPic = new JLabel(imageIcon);
        JButton jButton = new JButton( text: "Send image to server.");
        jFrame.add(jLabelPic);
        jFrame.add(jButton);
        jFrame.setVisible(true);
        jButton.addActionListener(new ActionListener() {
            @Override public void actionPerformed(ActionEvent event) {
                try{
                    OutputStream outputStream = socket.getOutputStream();
                    BufferedOutputStream bufferedOutputStream = new BufferedOutputStream(outputStream);
                    Image image = imageIcon.getImage();
                    BufferedImage bufferedImage = new BufferedImage(image.getWidth( observer: null), image.getHeight( observer: null), BufferedImage.TYPE_INT_RGB);
                    Graphics graphics = bufferedImage.createGraphics();
                    graphics.drawImage(image,  x: 0,  y: 0,  observer: null);
                    graphics.dispose();
                    ImageIO.write(bufferedImage,  formatName: "png", bufferedOutputStream);
                    bufferedOutputStream.close();
                    socket.close();
                } catch (IOException e) {
                    e.printStackTrace();
                }
            }
        });
    }
}
```

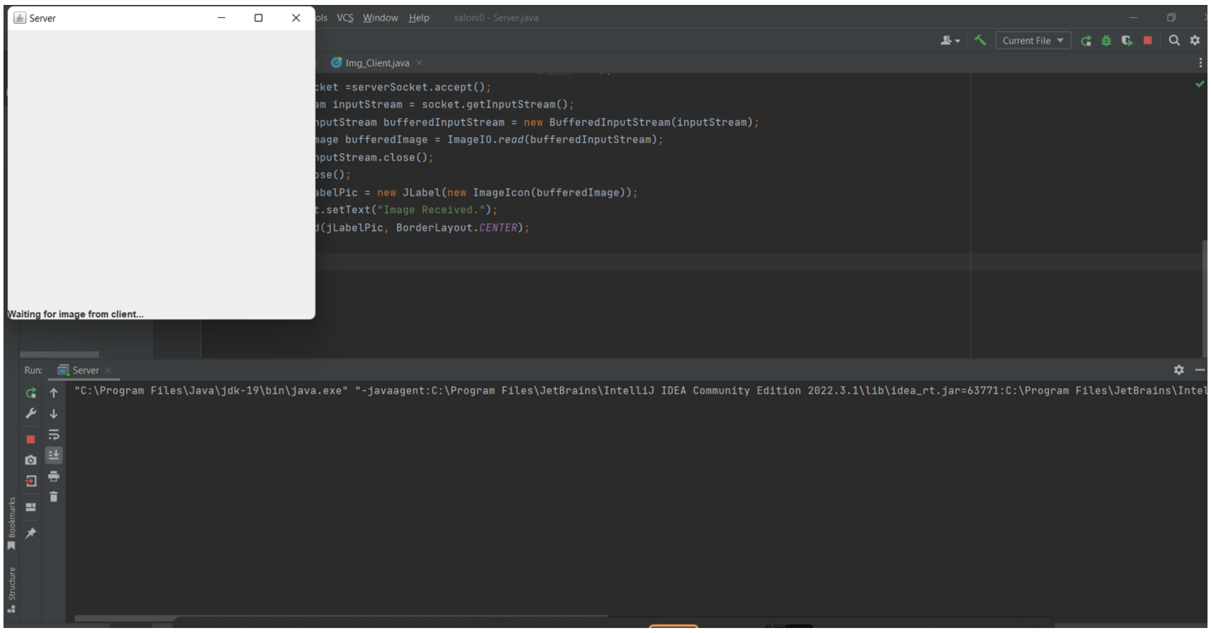
Server:


Server.java × Client.java ×

```
import javax.imageio.ImageIO;
import javax.swing.*;
import java.awt.*;
import java.awt.image.BufferedImage;
import java.io.BufferedInputStream;
import java.io.IOException;
import java.io.InputStream;
import java.net.ServerSocket;
import java.net.Socket;
no usages
public class Server {
    no usages
    public static void main(String[] args) throws IOException {
        JFrame jFrame = new JFrame( title: "Server");
        jFrame.setSize( width: 400,  height: 400);
        jFrame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        JLabel jLabelText = new JLabel( text: "Waiting for image from client...");
        jFrame.add(jLabelText, BorderLayout.SOUTH);
        jFrame.setVisible(true);
        ServerSocket serverSocket = new ServerSocket( port: 1234);
        Socket socket =serverSocket.accept();
        InputStream inputStream = socket.getInputStream();
        BufferedInputStream bufferedInputStream = new BufferedInputStream(inputStream);
        BufferedImage bufferedImage = ImageIO.read(bufferedInputStream);
        bufferedInputStream.close();
        socket.close();
        JLabel jLabelPic = new JLabel(new ImageIcon(bufferedImage));
        jLabelText.setText("Image Received.");
        jFrame.add(jLabelPic, BorderLayout.CENTER);
    }
}
```

Image-transfer (Outputs):

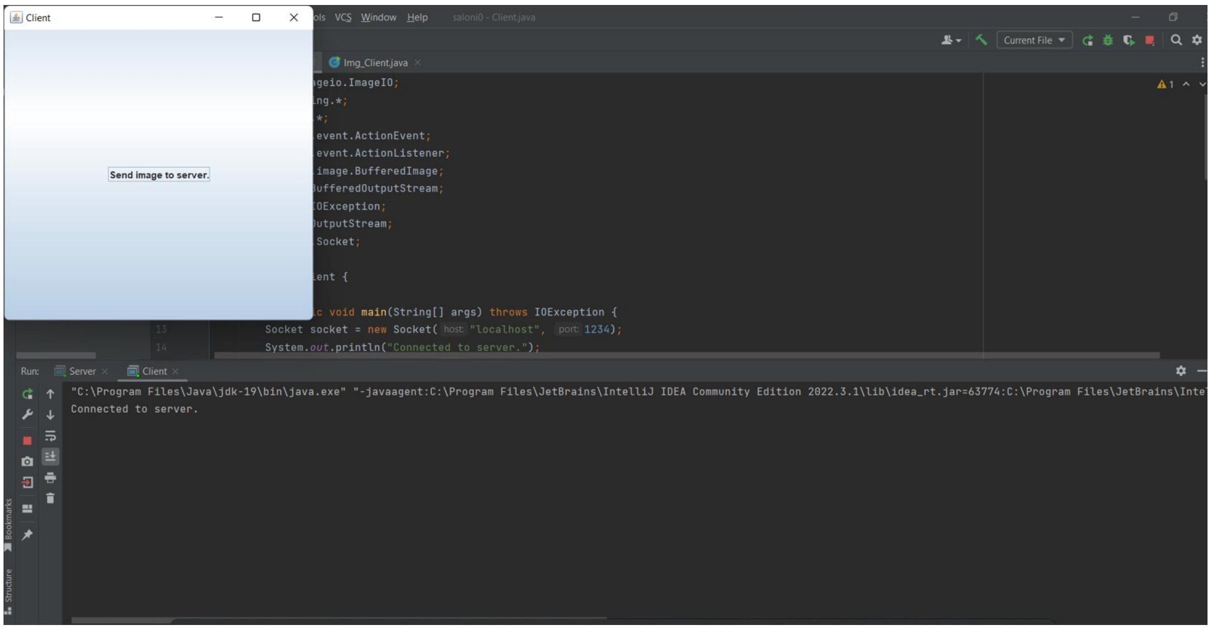
Server


Server
ols VCS Window Help saloni0 - Server.java
Current File
Img_Client.java
Waiting for image from client...
Run: Server

```
cket =serverSocket.accept();
am inputStream = socket.getInputStream();
nputStream bufferedInputStream = new BufferedInputStream(inputStream);
mage bufferedImage = ImageIO.read(bufferedInputStream);
nputStream.close();
ose();
abelPic = new JLabel(new ImageIcon(bufferedImage));
t.setText("Image Received.");
d(jLabelPic, BorderLayout.CENTER);
```

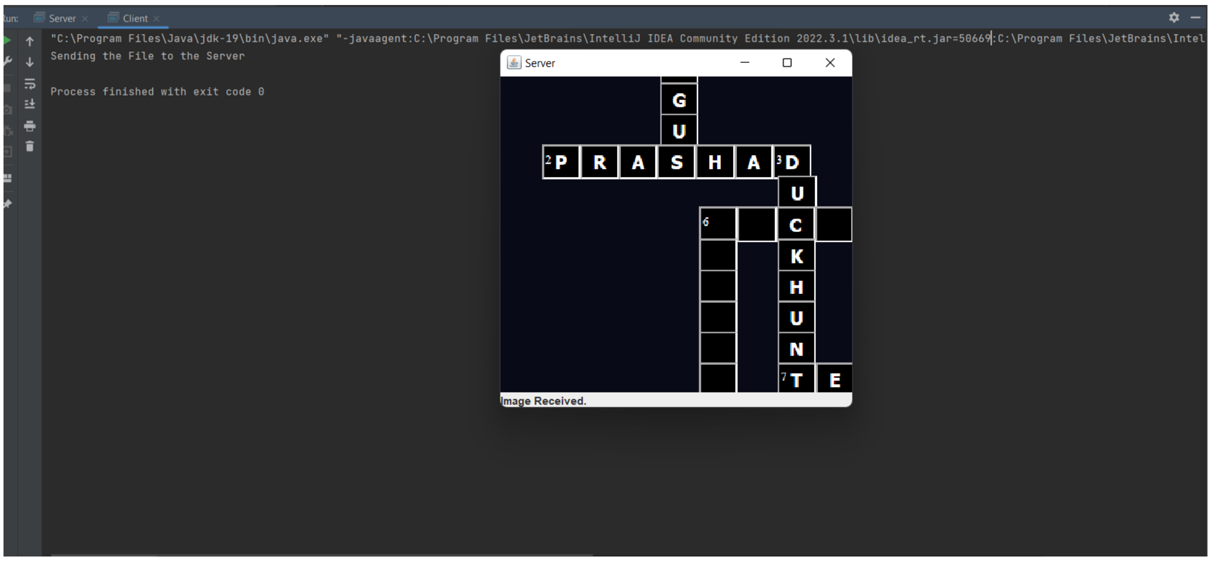
```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=63771:C:\Program Files\JetBrains\Intel
```

Client



```
ageio.ImageIO;
ing.*;
*;
event.ActionEvent;
event.ActionListener;
image.BufferedImage;
BufferedOutputStream;
IOException;
OutputStream;
Socket;

ent {

c void main(String[] args) throws IOException {
    Socket socket = new Socket( host: "localhost",  port: 1234);
    System.out.println("Connected to server.");
```

```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=63774:C:\Program Files\JetBrains\Inte
Connected to server.
```



```
"C:\Program Files\Java\jdk-19\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2022.3.1\lib\idea_rt.jar=50669:C:\Program Files\JetBrains\Intel
Sending the File to the Server

Process finished with exit code 0
```