

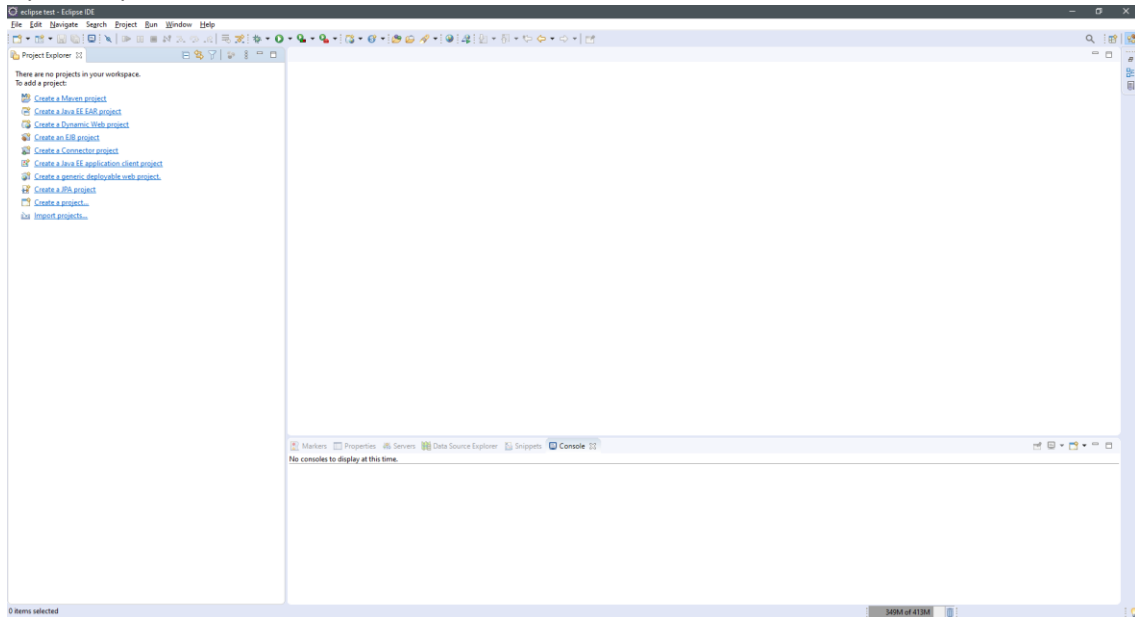
Name: Rajath Muthappa Kallichanda

Student ID: 1001724662

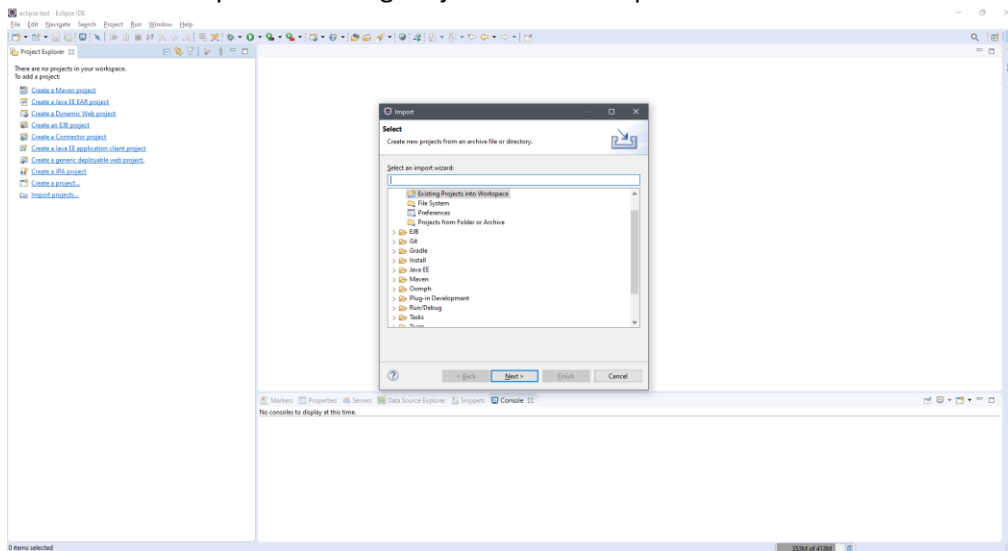
Steps to Run the Project

Steps to run the project.

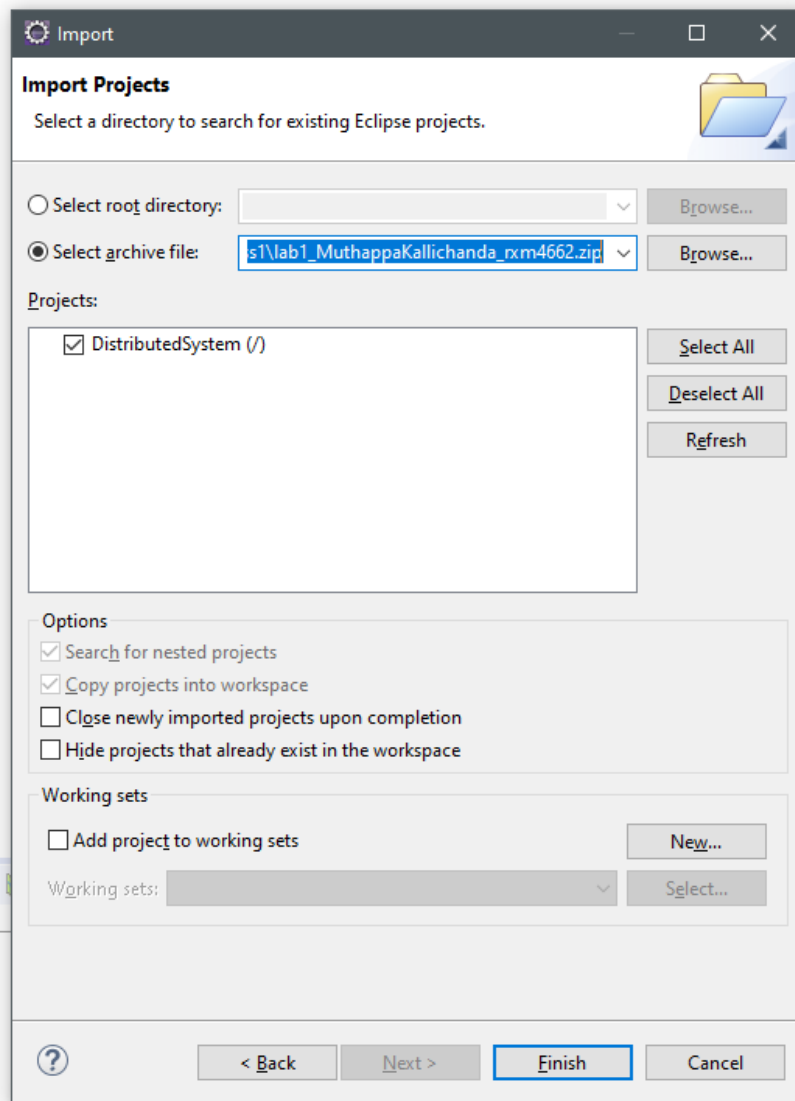
1. Open eclipse



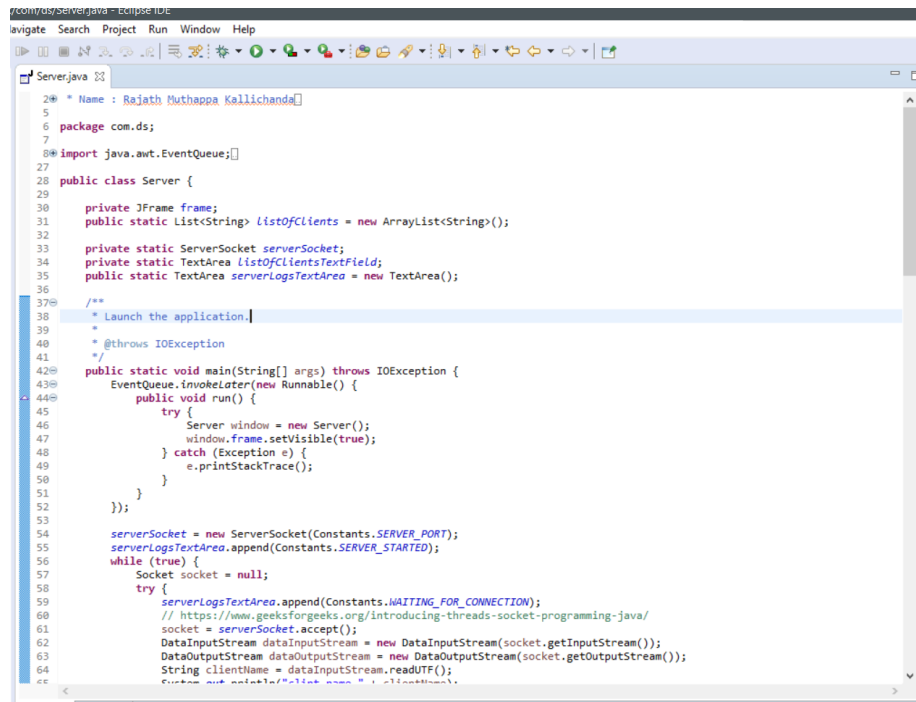
2. Click on File -> Import -> Existing Projects into Workspace



3. Select the archive file radio button and browse enter the path to the zip file



4. Click on finish.
5. Wait until the project is automatically built or select Project -> Build All
6. Open the Server.java file and click on run button or right click and select run as-> Java Application

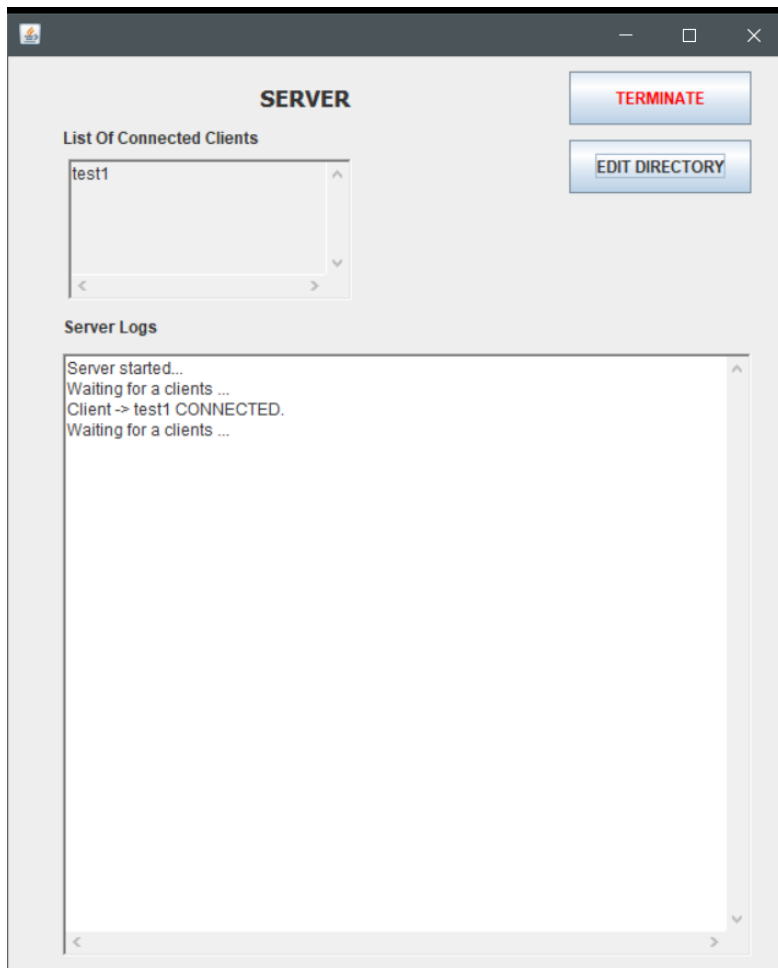


```
2 * Name : Rajath Muthappa Kollichanda
5
6 package com.ds;
7
8 import java.awt.EventQueue;
9
10 public class Server {
11
12     private JFrame frame;
13     public static List<String> listOfClients = new ArrayList<String>();
14
15     private static ServerSocket serverSocket;
16     private static TextArea listOfClientsTextField;
17     public static TextArea serverLogsTextArea = new TextArea();
18
19     /**
20      * Launch the application.
21      *
22      * @throws IOException
23      */
24     public static void main(String[] args) throws IOException {
25         EventQueue.invokeLater(new Runnable() {
26             public void run() {
27                 try {
28                     Server window = new Server();
29                     window.frame.setVisible(true);
30                 } catch (Exception e) {
31                     e.printStackTrace();
32                 }
33             }
34         });
35
36         serverSocket = new ServerSocket(Constants.SERVER_PORT);
37         serverLogsTextArea.append(Constants.SERVER_STARTED);
38         while (true) {
39             Socket socket = null;
40             try {
41                 serverLogsTextArea.append(Constants.WAITING_FOR_CONNECTION);
42                 // https://www.geeksforgeeks.org/introducing-threads-socket-programming-java/
43                 socket = serverSocket.accept();
44                 DataInputStream dataInputStream = new DataInputStream(socket.getInputStream());
45                 DataOutputStream dataOutputStream = new DataOutputStream(socket.getOutputStream());
46                 String clientName = dataInputStream.readUTF();
47                 listOfClients.add(clientName);
48                 listOfClientsTextField.setText(listOfClients.toString());
49                 dataOutputStream.writeUTF(clientName);
50                 dataOutputStream.flush();
51             } catch (Exception e) {
52                 e.printStackTrace();
53             }
54         }
55     }
56 }
```

7. Server Window will pop up and the server is up and running
8. Open ClientConnection.java and run the file as mentioned above.
9. Enter valid client name and click on connect button.
10. On validation, a new window would open for user to enter the commands that would be sent to the server for processing.

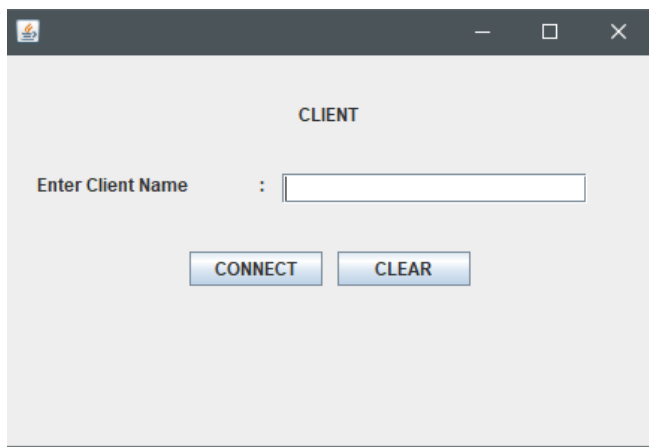
NOTE: Please start the server before trying to establish a client connection.

Server Window:



Click on terminate to terminate the server.

Client Connection window:



Enter the client name and click on connect to establish a connection. Click on clear to clear the text area.

Client Window after connection is established:

The screenshot shows a client window titled "Client : test1". The interface is divided into two main sections. The left section contains several form fields and buttons for directory management: "Enter the Directory name to be created" with a "CREATE DIRECTO..." button; "Enter the Directory name/path to be deleted" with a "DELETE DIRECTORY" button; "Enter the Directory name to rename" and "Enter the new Directory name." with a "RENAME DIRECTO..." button; "Enter the Directory name / path to be moved." and "Enter the new Directory path." with a "MOVE DIRECTORY" button; "SYNC SERVER DIRECTORY" and "DE-SYNC SERVER DIRECT..." buttons; and "BROWSE" and "DISCONNECT" buttons at the bottom. The right section is titled "Directory View" and features a "Refresh" button. Below the button is a text area displaying "Files from main directory : .roottest1" surrounded by asterisks, with a vertical scrollbar on the right.

1. Create Directory.

Enter the directory name in the text area and click on CREATE_DIRECTORY. This will send the instructions to the server and on successful validation of the folder name, a new folder would be created, and a response would be sent back to the client. Appropriate message would be displayed to the client if the operation was successful or not.

2. Delete Directory.

Enter the directory name in the text area and click on DELETE_DIRECTORY. This will send the instructions to the server. The server checks if the folder exists or not. If it exists it will delete else it will send appropriate response back to the client.

3. Rename Directory.

Enter the Directory whose name has to be change. In the other text box enter the new name. Click on RENAME_DIRECTORY. This will send the instructions to the server and based on the results, appropriate responses would be sent back to the client.

4. Move Directory.

Enter the Directory name to be moved in the first text box, and the new location in the second text area and Click on MOVE_DIRECTORY. This will send the instruction to the server and appropriate response will be sent back to the client based on the outcome of the operation.

5. Disconnect.

Click on the DISCONNECT button to terminate the session with the server.

6. Directory View.

Directory view would list all the directory in the clients Directory. This gets updated dynamically and you may also click on refresh to get the latest folder structure view.

NOTE: if there are any compilation errors due to missing jars, please add the jars manually to the project build path.

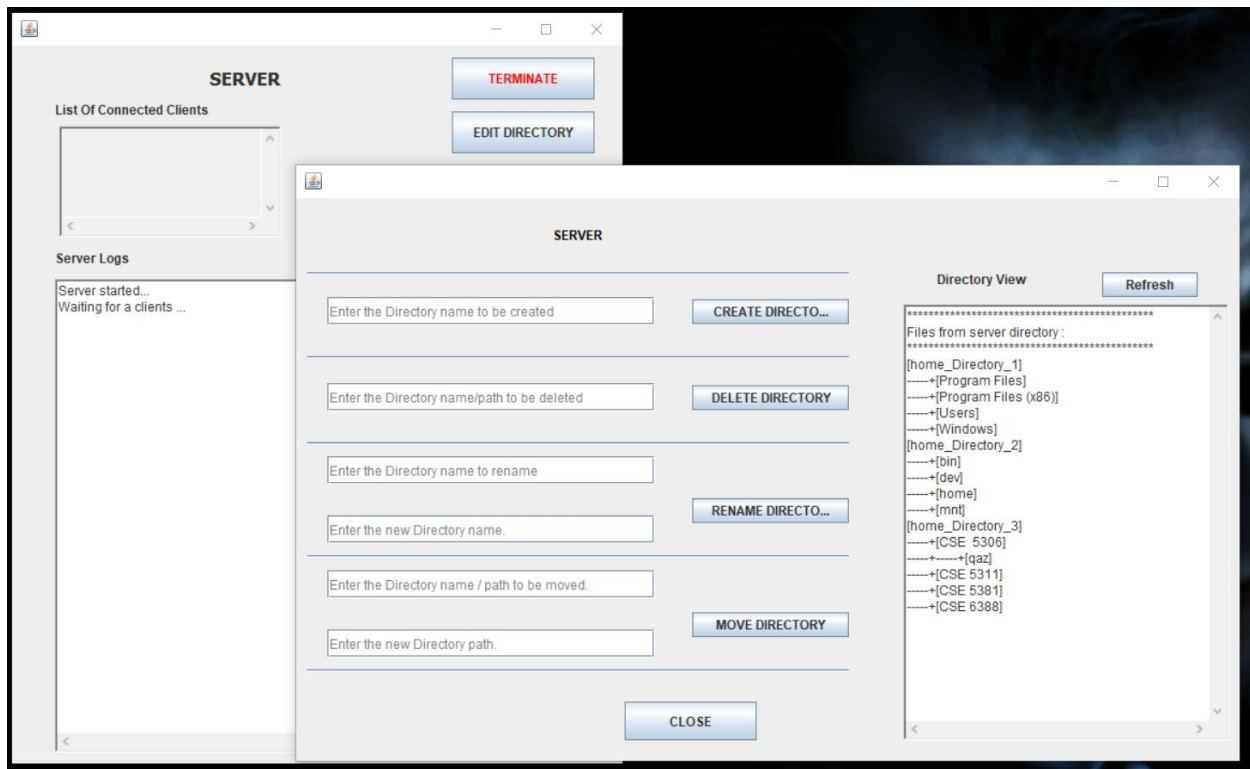
Right click on the project -> build path -> configure build path -> add external jars.

Please navigate to the extracted folder to find, “commons-io-2.8.0.jar” and add it to the class path before running the project

PART 2:

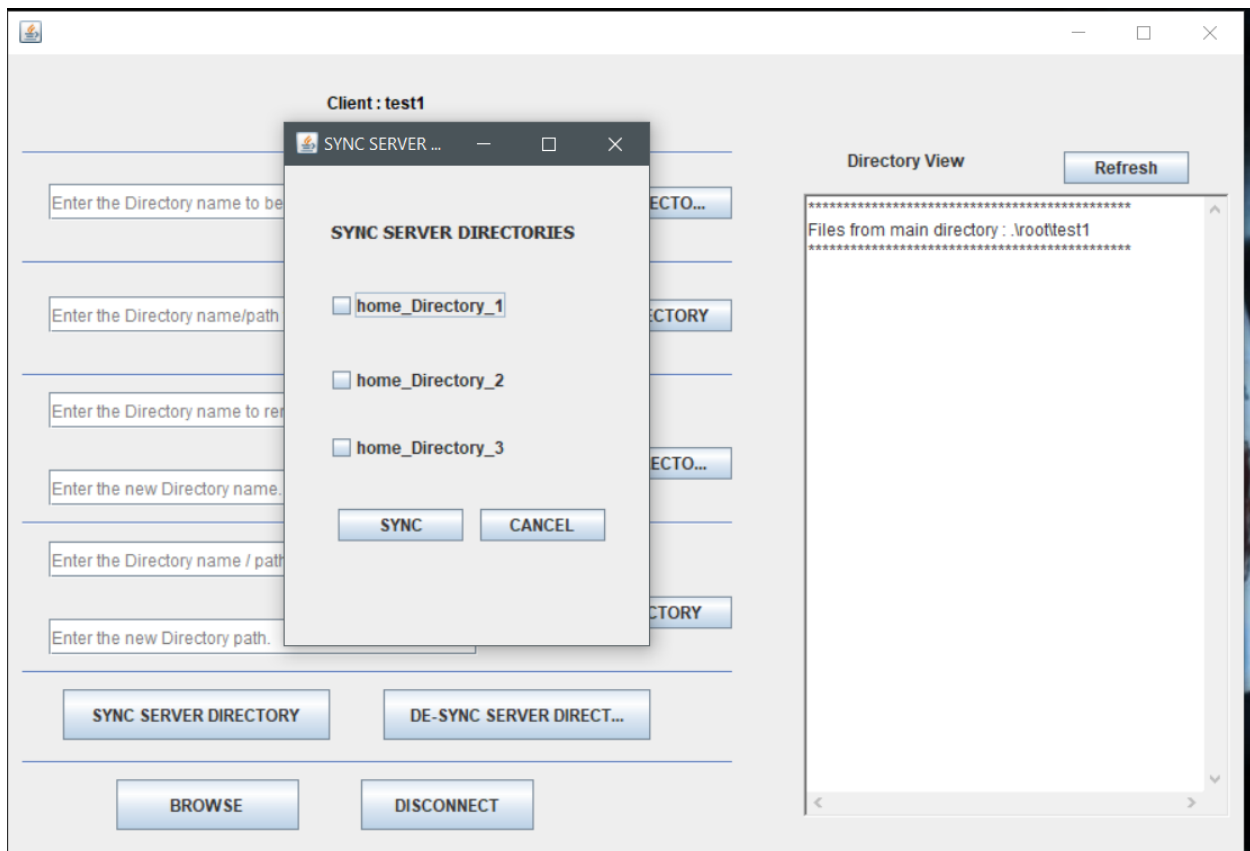
Server Window:

Click on the edit directory to modify the Server side directory.

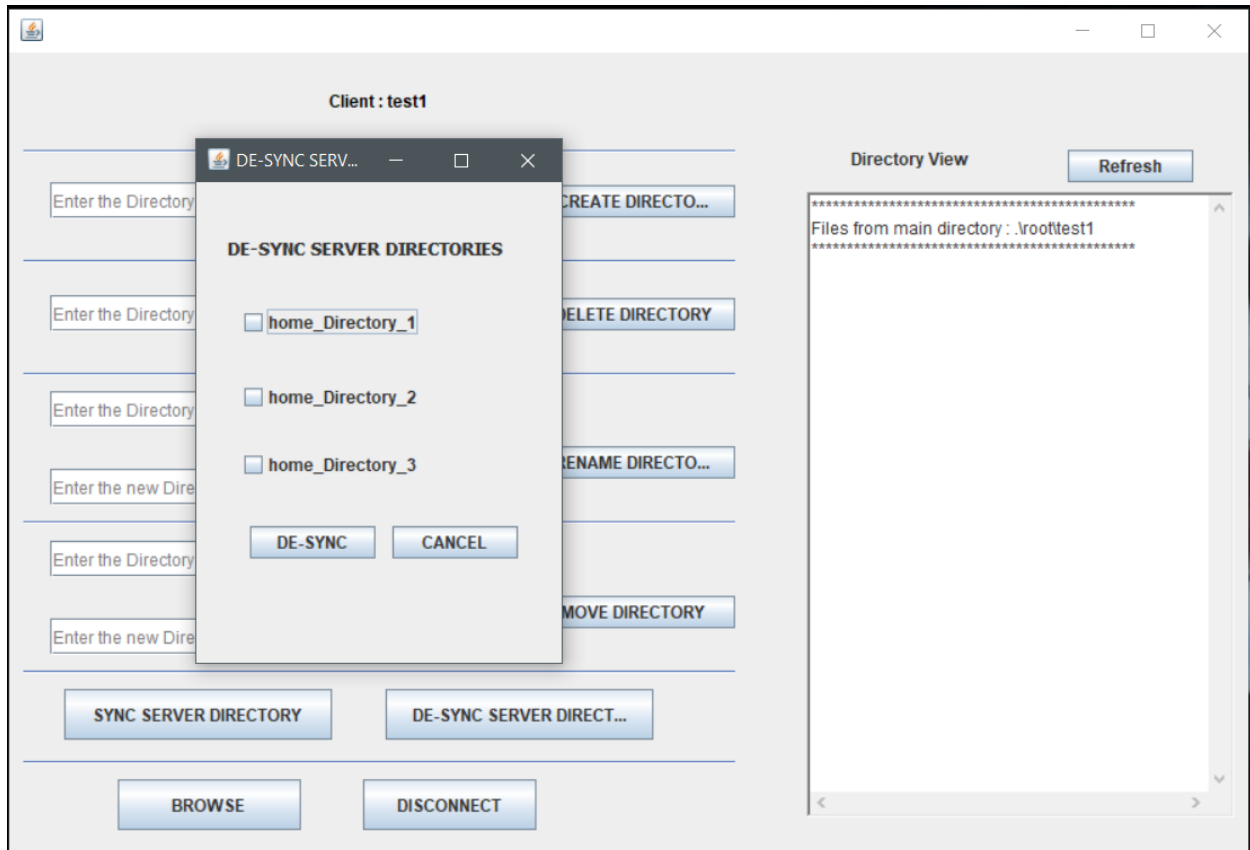


Client Window:

Click on the “SYNC SERVER DIRECTORY” button to access the list of directories available for synchronization. Click on Sync button once the necessary directories are selected. This would create a local directory in clients root folder along with an Identifier that is dynamically assigned to it when a request to SYNC is made.



Click on the “DE-SYNC SERVER DIRECTORY” button to access the list of directories available for de-synchronization. Click on De-Sync button once the necessary directories are selected. This would de-sync/delete the local directory copy in clients root folder.



Any changes made to the to the SERVER directory is automatically synced. CREATE, DELETE, RENAME and MOVE events would be automatically synced along all shared folders in clients local directories. MOVE folder from one root directory to another root directory has a special check. If the client has not made a copy of the new Directory to which the server is moving the file to, then the file is deleted from the current local directory and no copy of it would be available to the client. Synchronization of folders would take place even if the client is offline. If the SERVER changes are not being reflected please click on the refresh button for the directory tree structure to update to the latest changes.

Part 3

Part 3 of the project involves in performing UNDO operation that are done by the user at the SERVER end. The UNDO operation would also remove the associated logs from the server logs to maintain consistency. The Server UI is provided with a text box to enter the logged command to UNDO the changes. The logged statements that could be used UNDO operations has (CMD) in the beginning. These logged statements should be used to perform UNDO operation.

Steps to perform UNDO:

Step 1: Perform an operation.

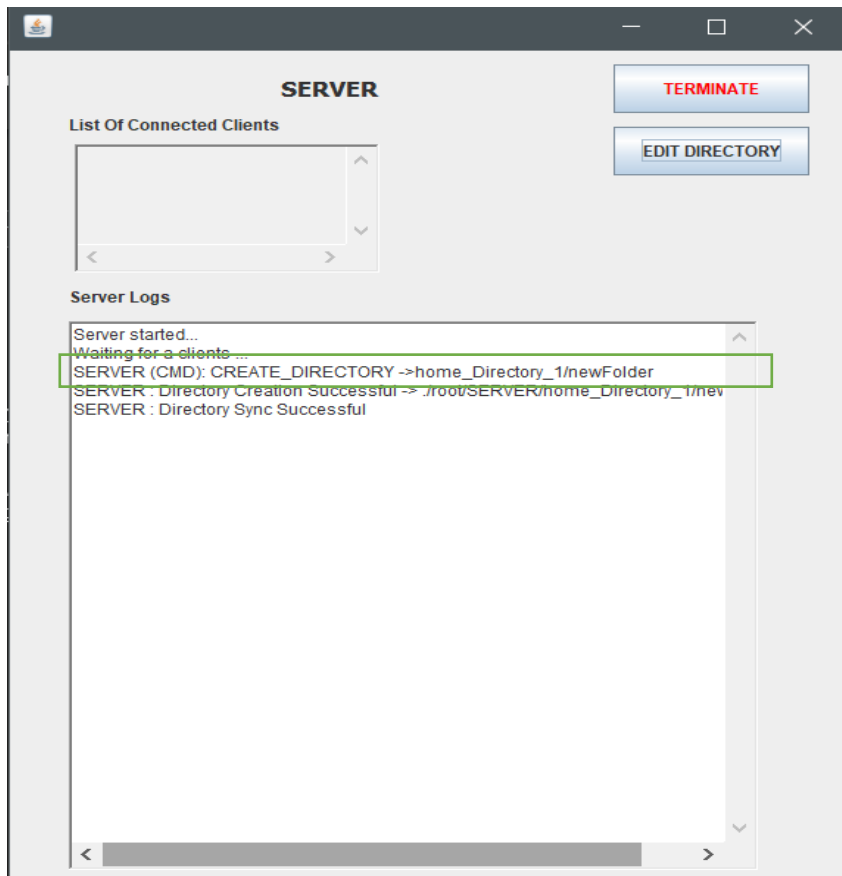
The screenshot displays a web application window titled "SERVER". The interface is divided into two main sections. On the left, there are six rows of input fields and buttons for directory management:

- Row 1: "Enter the Directory name to be created" followed by a "CREATE DIRECTO..." button.
- Row 2: "Enter the Directory name/path to be deleted" followed by a "DELETE DIRECTORY" button.
- Row 3: "Enter the Directory name to rename" followed by a "RENAME DIRECTO..." button.
- Row 4: "Enter the new Directory name." (input field only).
- Row 5: "Enter the Directory name / path to be moved." followed by a "MOVE DIRECTORY" button.
- Row 6: "Enter the new Directory path." (input field only).
- Row 7: "Enter Logged operation to Undo" followed by an "UNDO" button.

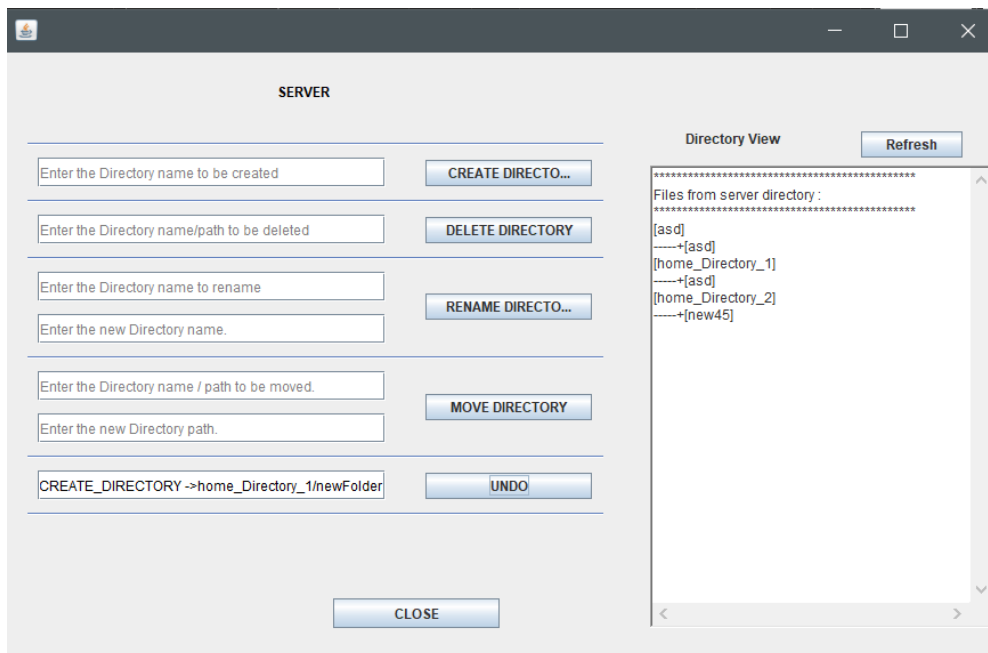
At the bottom center of the left section is a "CLOSE" button. On the right side, there is a "Directory View" section with a "Refresh" button. Below the button is a text area showing the output of a directory listing:

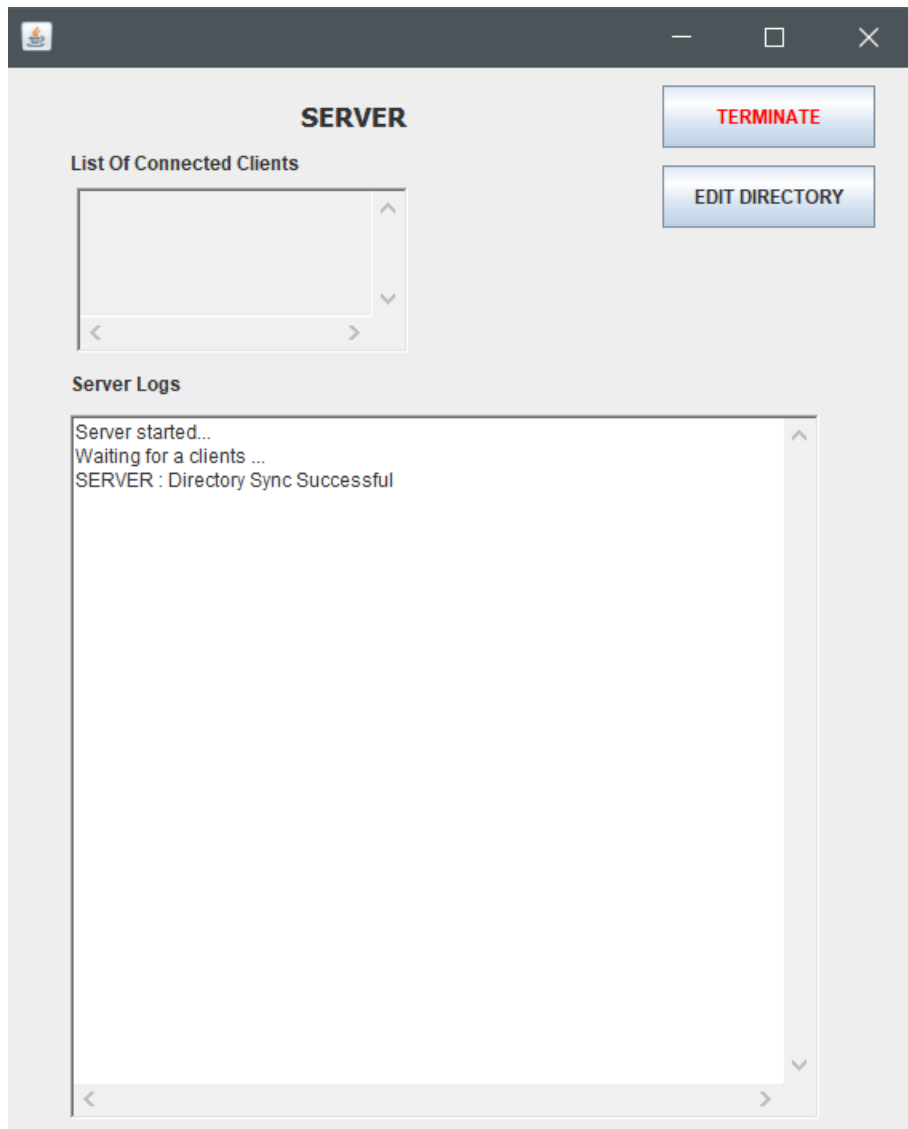
```
*****
Files from server directory :
*****
[asd]
-----+[asd]
[home_Directory_1]
-----+[asd]
[home_Directory_2]
-----+[new45]
```

Step 2: Select the logged statement that starts with (CMD)



Step 3: Enter the logged command and click on UNDO.





The operation is performed and the log from the server logs are also cleared.

CITATIONS:

<https://www.geeksforgeeks.org/introducing-threads-socket-programming-java/>

<https://stackoverflow.com/questions/17018857/how-to-call-jframe-from-another-java-class>

<https://stackoverflow.com/questions/16213836/java-swing-jtextfield-set-placeholder>

<https://stackoverflow.com/questions/48339967/how-to-rename-the-folder-in-java>

<https://www.studytrails.com/java-io/file-copying-and-moving-deleting/>

<https://stackoverflow.com/questions/20281835/how-to-delete-a-folder-with-files-using-java>