

Untitled-1

```
1 import java.io.*;
2 import java.net.*;
3 import java.util.*;
4
5 public class FileServer {
6     private static final String FILE_DIRECTORY = "server_files"; // Directory to store files
7
8     public static void main(String[] args) {
9         try {
10             ServerSocket serverSocket = new ServerSocket(12345);
11             System.out.println("File server started. Waiting for clients...");
12
13             // Create the directory if it doesn't exist
14             File directory = new File(FILE_DIRECTORY);
15             if (!directory.exists()) {
16                 directory.mkdir();
17             }
18
19             while (true) {
20                 Socket clientSocket = serverSocket.accept();
21                 System.out.println("Client connected.");
22
23                 new ClientHandler(clientSocket).start();
24             }
25         } catch (IOException e) {
26             e.printStackTrace();
27         }
28     }
29
30     private static class ClientHandler extends Thread {
31         private Socket clientSocket;
32
33         public ClientHandler(Socket socket) {
34             this.clientSocket = socket;
35         }
36
37         public void run() {
38             try (
39                 DataInputStream dis = new DataInputStream(clientSocket.getInputStream());
40                 DataOutputStream dos = new DataOutputStream(clientSocket.getOutputStream());
41             ) {
42                 String command = dis.readUTF();
43
44                 switch (command) {
45                     case "UPLOAD":
46                         uploadFile(dis, dos);
47                         break;
48                     case "DOWNLOAD":
```

```
49         downloadFile(dis, dos);
50         break;
51     case "LIST":
52         listFiles(dos);
53         break;
54     default:
55         dos.writeUTF("Invalid command.");
56     }
57 } catch (IOException e) {
58     e.printStackTrace();
59 }
60 }
61
62 private void uploadFile(DataInputStream dis, DataOutputStream dos) throws IOException {
63     String fileName = dis.readUTF();
64     long fileSize = dis.readLong();
65     File file = new File(FILE_DIRECTORY + "/" + fileName);
66
67     try (FileOutputStream fos = new FileOutputStream(file)) {
68         byte[] buffer = new byte[4096];
69         int read;
70         long totalRead = 0;
71         while ((read = dis.read(buffer, 0, Math.min(buffer.length, (int) (fileSize -
totalRead)))) > 0) {
72             fos.write(buffer, 0, read);
73             totalRead += read;
74         }
75         System.out.println("File " + fileName + " uploaded successfully.");
76         dos.writeUTF("File uploaded successfully.");
77     }
78 }
79
80 private void downloadFile(DataInputStream dis, DataOutputStream dos) throws IOException
81 {
82     String fileName = dis.readUTF();
83     File file = new File(FILE_DIRECTORY + "/" + fileName);
84
85     if (file.exists()) {
86         dos.writeUTF("OK");
87         dos.writeLong(file.length());
88
89         try (FileInputStream fis = new FileInputStream(file)) {
90             byte[] buffer = new byte[4096];
91             int read;
92             while ((read = fis.read(buffer)) > 0) {
93                 dos.write(buffer, 0, read);
94             }
95             System.out.println("File " + fileName + " downloaded successfully.");
96         } else {
```

```
97         dos.writeUTF("File not found.");
98     }
99 }
100
101 private void listFiles(DataOutputStream dos) throws IOException {
102     File directory = new File(FILE_DIRECTORY);
103     String[] files = directory.list();
104
105     if (files != null && files.length > 0) {
106         dos.writeUTF("OK");
107         dos.writeInt(files.length);
108         for (String file : files) {
109             dos.writeUTF(file);
110         }
111     } else {
112         dos.writeUTF("No files available.");
113     }
114 }
115 }
116 }
117 }
```

Untitled-2

```
1 import java.io.*;
2 import java.net.*;
3
4 public class FileClient {
5     public static void main(String[] args) {
6         try (Socket socket = new Socket("localhost", 12345);
7             DataInputStream dis = new DataInputStream(socket.getInputStream());
8             DataOutputStream dos = new DataOutputStream(socket.getOutputStream());
9             BufferedReader br = new BufferedReader(new InputStreamReader(System.in))) {
10
11             System.out.println("Connected to file server.");
12
13             while (true) {
14                 System.out.println("\nEnter a command (UPLOAD, DOWNLOAD, LIST, EXIT):");
15                 String command = br.readLine();
16
17                 if ("EXIT".equalsIgnoreCase(command)) {
18                     break;
19                 }
20
21                 dos.writeUTF(command);
22
23                 switch (command.toUpperCase()) {
24                     case "UPLOAD":
25                         uploadFile(dis, dos, br);
26                         break;
27                     case "DOWNLOAD":
28                         downloadFile(dis, dos, br);
29                         break;
30                     case "LIST":
31                         listFiles(dis);
32                         break;
33                     default:
34                         System.out.println("Invalid command.");
35                 }
36             }
37         } catch (IOException e) {
38             e.printStackTrace();
39         }
40     }
41
42     private static void uploadFile(DataInputStream dis, DataOutputStream dos, BufferedReader
br) throws IOException {
43         System.out.print("Enter the file path to upload: ");
44         String filePath = br.readLine();
45         File file = new File(filePath);
46
47         if (file.exists() && !file.isDirectory()) {
```

```
48         dos.writeUTF(file.getName());
49         dos.writeLong(file.length());
50
51         try (FileInputStream fis = new FileInputStream(file)) {
52             byte[] buffer = new byte[4096];
53             int read;
54             while ((read = fis.read(buffer)) > 0) {
55                 dos.write(buffer, 0, read);
56             }
57         }
58         System.out.println("Server response: " + dis.readUTF());
59     } else {
60         System.out.println("File not found.");
61     }
62 }
63
64 private static void downloadFile(DataInputStream dis, DataOutputStream dos, BufferedReader
br) throws IOException {
65     System.out.print("Enter the file name to download: ");
66     String fileName = br.readLine();
67     dos.writeUTF(fileName);
68
69     String response = dis.readUTF();
70     if ("OK".equals(response)) {
71         long fileSize = dis.readLong();
72         File file = new File("client_files/" + fileName);
73
74         try (FileOutputStream fos = new FileOutputStream(file)) {
75             byte[] buffer = new byte[4096];
76             int read;
77             long totalRead = 0;
78             while ((read = dis.read(buffer, 0, Math.min(buffer.length, (int) (fileSize -
totalRead)))) > 0) {
79                 fos.write(buffer, 0, read);
80                 totalRead += read;
81             }
82             System.out.println("File downloaded successfully.");
83         }
84     } else {
85         System.out.println("Server response: " + response);
86     }
87 }
88
89 private static void listFiles(DataInputStream dis) throws IOException {
90     String response = dis.readUTF();
91     if ("OK".equals(response)) {
92         int fileCount = dis.readInt();
93         System.out.println("Files available on server:");
94         for (int i = 0; i < fileCount; i++) {
95             System.out.println("- " + dis.readUTF());
```

```
96         }
97     } else {
98         System.out.println("Server response: " + response);
99     }
100 }
101 }
102
```