Gebze Technical University

CSE 344

System Programming

Spring 2024

Midterm Report

Buket Gençer

210104004298

Introduction

The primary goal of this assignment is to design and implement a concurrent file access system that allows multiple clients to interact with a file server. This system will enable clients to connect to the server, access and modify files, and perform operations like uploading and downloading files. Additionally, the system must ensure data consistency and prevent data corruption through effective use of synchronization mechanisms.

The server-client architecture is made up of two essential parts: the server and the client. The server handles incoming connections from clients, processes their requests, and delivers the necessary responses. On the other side, the client establishes a connection with the server and issues commands, to which it receives replies from the server.

Managing Multiple Client Connections

The server can handle a specified maximum number of clients simultaneously. This is achieved by:

- Forking a new process for each client connection: This ensures that each client has a dedicated process handling its requests, thus isolating sessions.
- Dynamic Client Management: The server keeps track of active clients using a PID array. This array helps manage client connections dynamically, allowing new connections as others are terminated.

Prevention of Race Conditions

Race conditions are prevented through:

- Mutual Exclusion: Semaphore mechanisms are used to control access to shared resources. This ensures that file operations by one process are completed before another can begin, maintaining data consistency.
- File Locking: Implemented at the server side to prevent simultaneous write operations on the same file, which could lead to data corruption.

Signal Handling

Proper signal handling ensures that the server can gracefully shut down in response to interruptions like SIGINT or custom signals from clients:

- Signal Handlers**:** These are set up to catch and respond to various signals within the server, ensuring that all child processes are terminated correctly and that resources are cleaned up before the server shuts down.

Makefile

```
# Compiler and linker configurations
CC = gcc
CFLAGS = -Wall -g
LDFLAGS =

# Define targets
all: clientx serverx

# Client and server executables
clientx: clientx.o logfile.o
    $(CC) $(LDFLAGS) -o $@ $^

serverx: serverx.o logfile.o
    $(CC) $(LDFLAGS) -o $@ $^

# Object files
clientx.o: clientx.c clientx.h logfile.h
    $(CC) $(CFLAGS) -c clientx.c

serverx.o: serverx.c serverx.h logfile.h
    $(CC) $(CFLAGS) -c serverx.c

logfile.o: logfile.c logfile.h
    $(CC) $(CFLAGS) -c logfile.c

# Clean old builds
clean:
    rm -f *.o clientx serverx

```

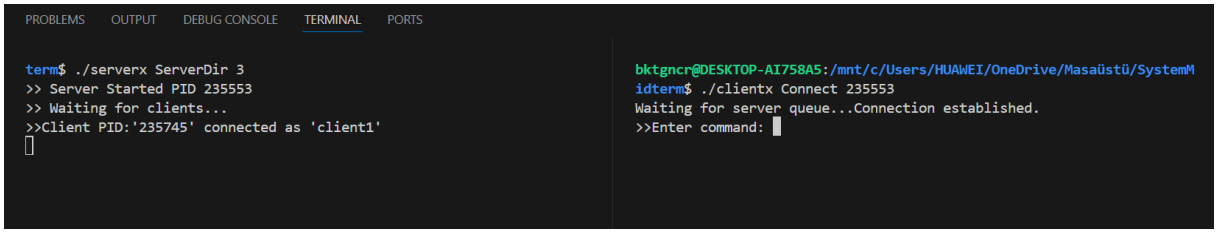
TESTING AND RESULT:

1) Connecting Clients:

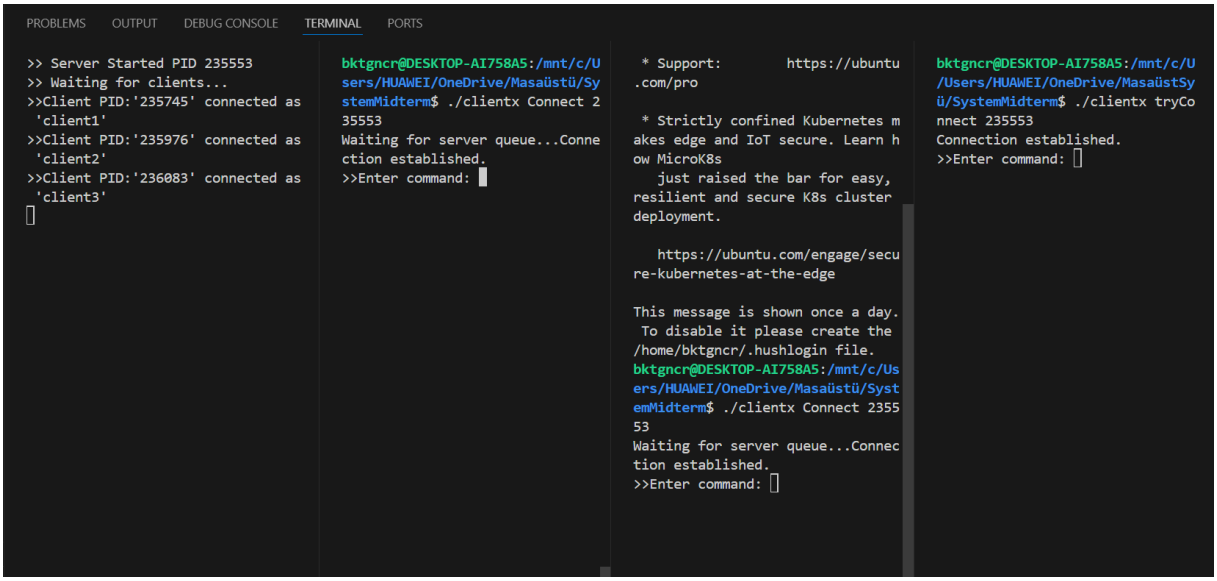
```
bktgncr@DESKTOP-AI758A5:/mnt/c/Users/HUAWEI/OneDrive/Masaüstü/SystemMid
term$ ./serverx ServerDir 3
>> Server Started PID 235553
>> Waiting for clients...
```

Connection examples:

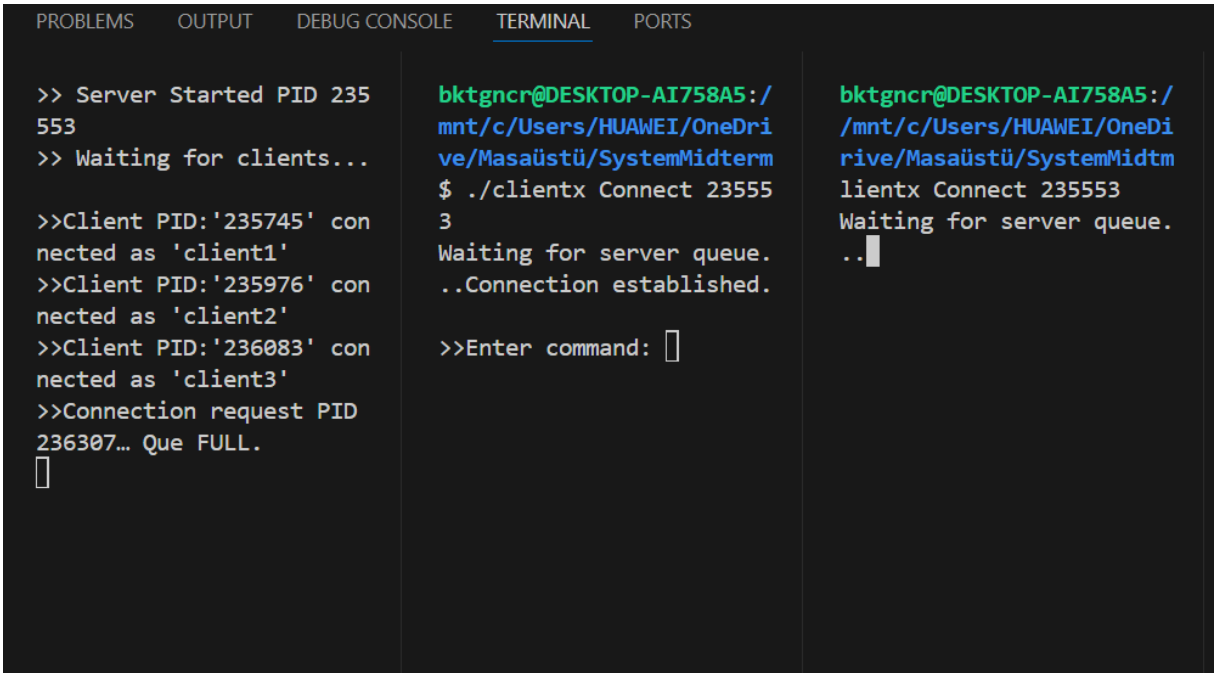
First client connect to server:



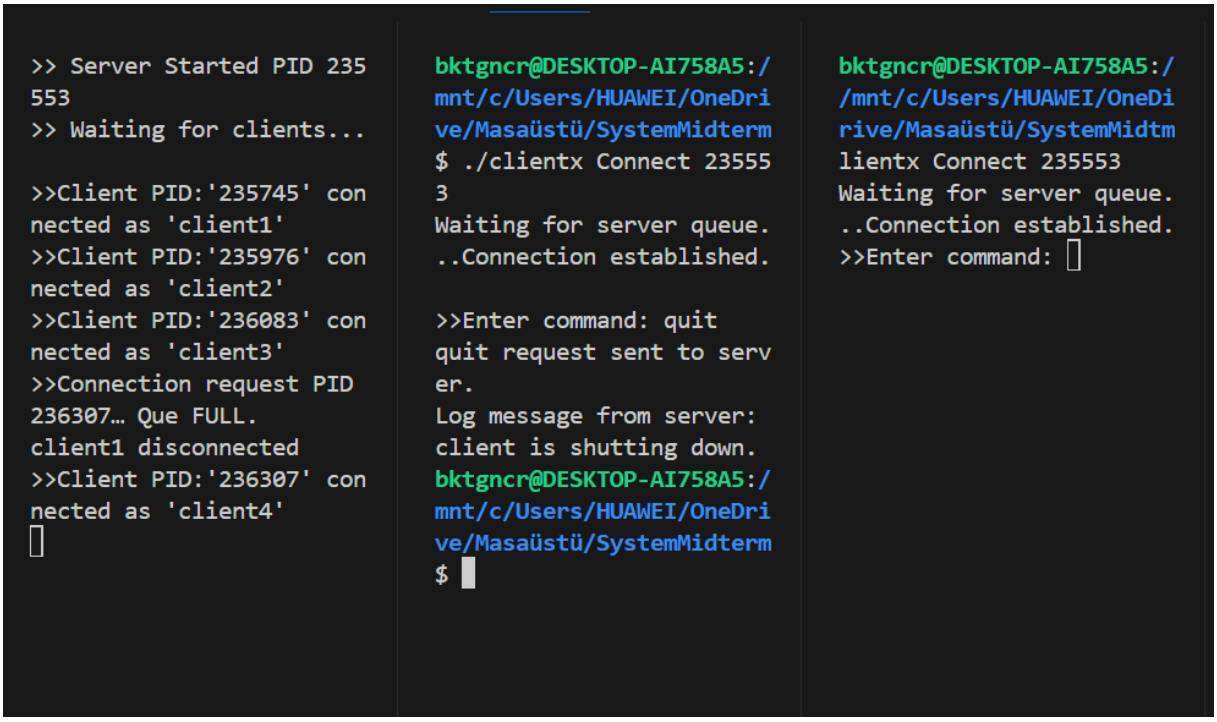
All possible connections:



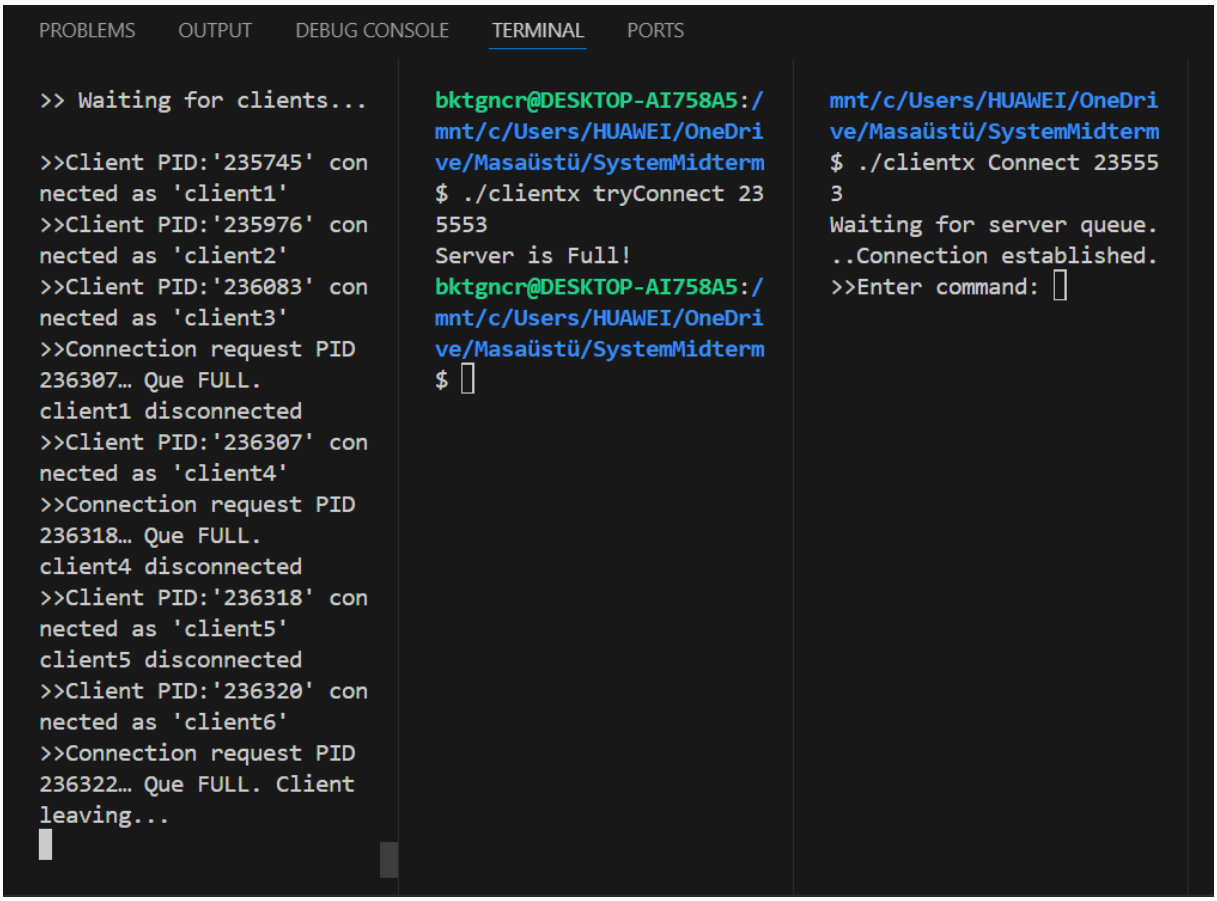
When the que full:



When a client disconnects, any clients waiting for a connection are then allowed to connect to the server.

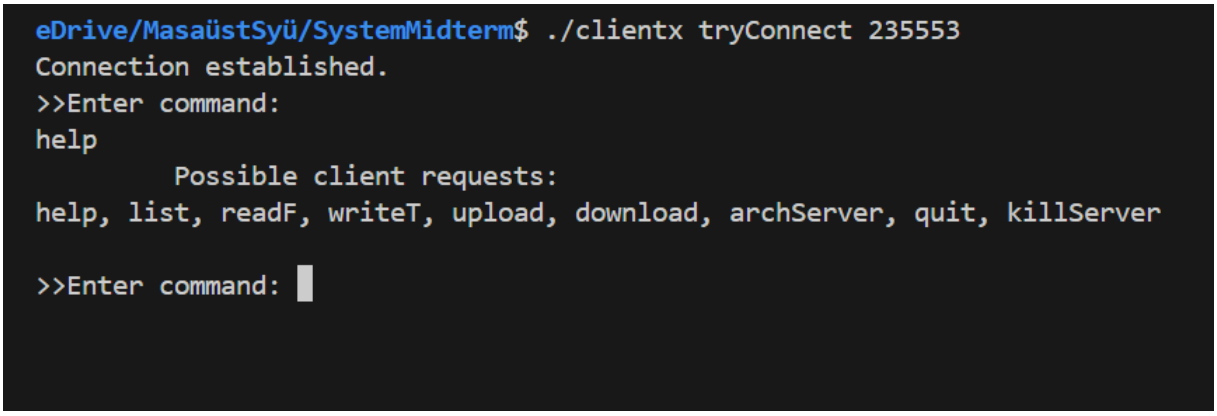


tryConnect when que is full:



Commends:

-help



Some version of help commend:

```
help
         Possible client requests:
help, list, readF, writeT, upload, download, archServer, quit, killServer

>>Enter command: help list
Usage: list
Display the list of files in Servers directory
(also displays the list received from the Server)
>>Enter command: help readF
Usage: readF <file> <line #>
Display the # line of the <file>, if no line number is given the whole cont
ents of the file is requested
>>Enter command: help writeT
Usage: writeT <file> <line #> <string>
Write the content of "string" to the #th line the <file>
 if the line # is not given writes to the end of file.
If the file does not exists in Servers directory
 creates and edits the file at the same time
>>Enter command: help upload
Usage: upload <file>
Uploads the file from the current working directory of client to the Server
s directory
>>Enter command: help download
Usage: download <file>
Request to receive <file> from Servers directory to client side
>>Enter command: help archServer
Usage: archServer <fileName>
Collect all the files currently available on the the Server side
 and store them in the <filename>.tar archive
>>Enter command: help quit
Usage: quit
Send write request to Server side log file and quits
>>Enter command: help killServer
Usage: killServer
Sends a kill request to the Server
>>Enter command:
```

-list

```
>>Enter command: list
Files in server directory:
all_logs.txt
log_235745.txt
log_235976.txt
log_236083.txt
log_236307.txt
log_236318.txt
log_236320.txt
log_236322.txt
>>Enter command:
```

```
>>Enter command: readF all_logs.txt
File content:
>> Server Started PID 235553
>> Waiting for clients...
>>Client PID:'235745' connected as 'client1'
>>Client PID:'235976' connected as 'client2'
>>Client PID:'236083' connected as 'client3'
>>Connection request PID 236307… Que FULL.
>>Client PID:'235745' connected as 'client1'
client1 disconnected
>>Client PID:'236307' connected as 'client4'
>>Connection request PID 236318… Que FULL.
>>Client PID:'236307' connected as 'client4'
client4 disconnected
>>Client PID:'236318' connected as 'client5'
>>Client PID:'236318' connected as 'client5'
client5 disconnected
>>Client PID:'236320' connected as 'client6'
>>Connection request PID 236322… Que FULL. Client leaving...
>>Client PID:'236083' connected as 'client3'
>>Client PID:'236083' connected as 'client3'

>>Enter command: readF all_logs.txt 5
File content:
>>Client PID:'236083' connected as 'client3'

>>Enter command:
```

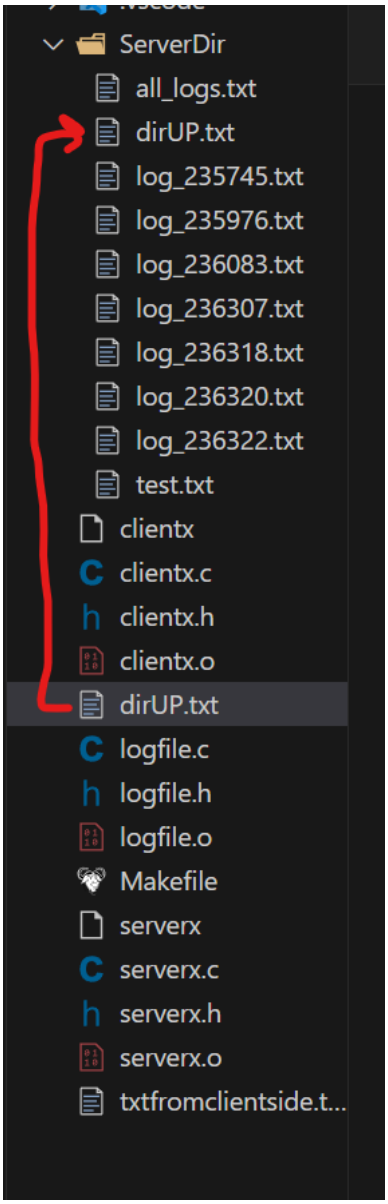
writeT:

```
>>Enter command: readF test.txt
File content:
first line
second line
>>Enter command: writeT test.txt 3 thirdline
Write successful.
>>Enter command: readF test.txt
File content:
first line
second line

>>Enter command: writeT test.txt 4 thirdline
Write successful.
>>Enter command: readF test.txt
File content:
first line
second line
thirdline

>>Enter command:
```

Upload:

```
>>Enter command: upload dirUP.txt
Server response:
File uploaded successfully. Transferred 5 bytes.
>>Enter command:
```

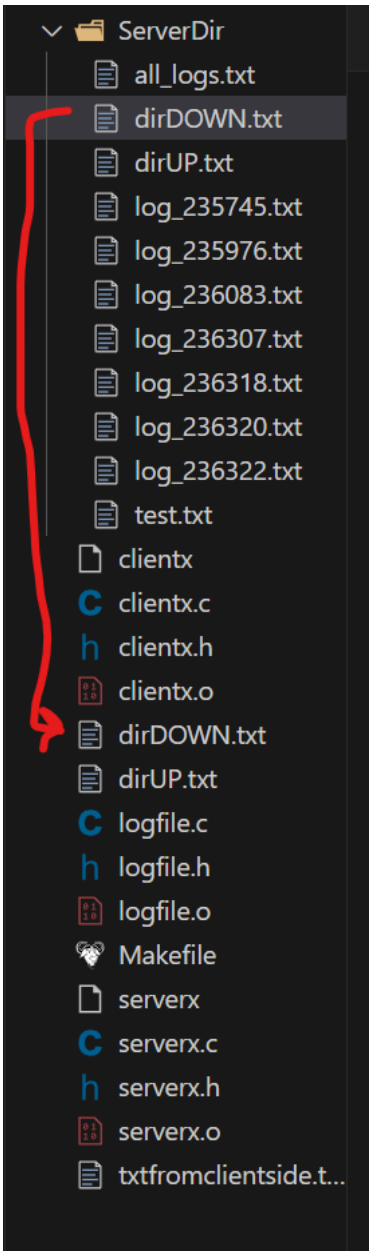
Result of upload:



Download:

```
>>Enter command: download dirDOWN.txt
Server response:
File downloaded successfully. Transferred 6 bytes.
>>Enter command:
```

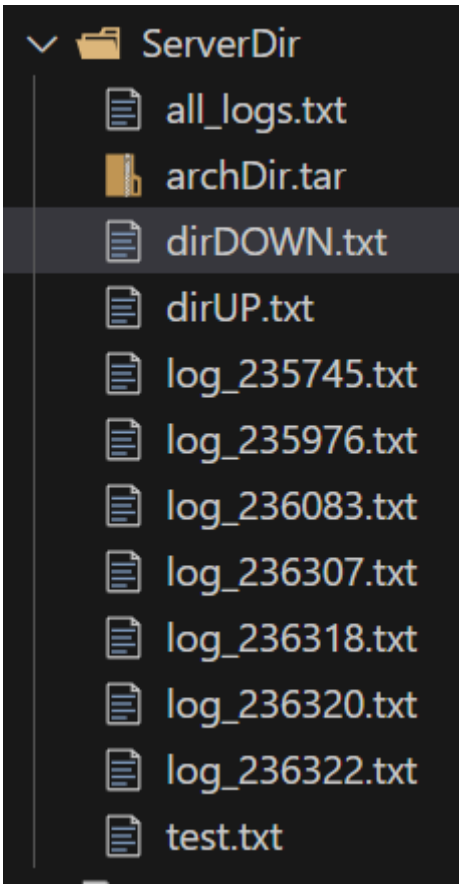
Result:



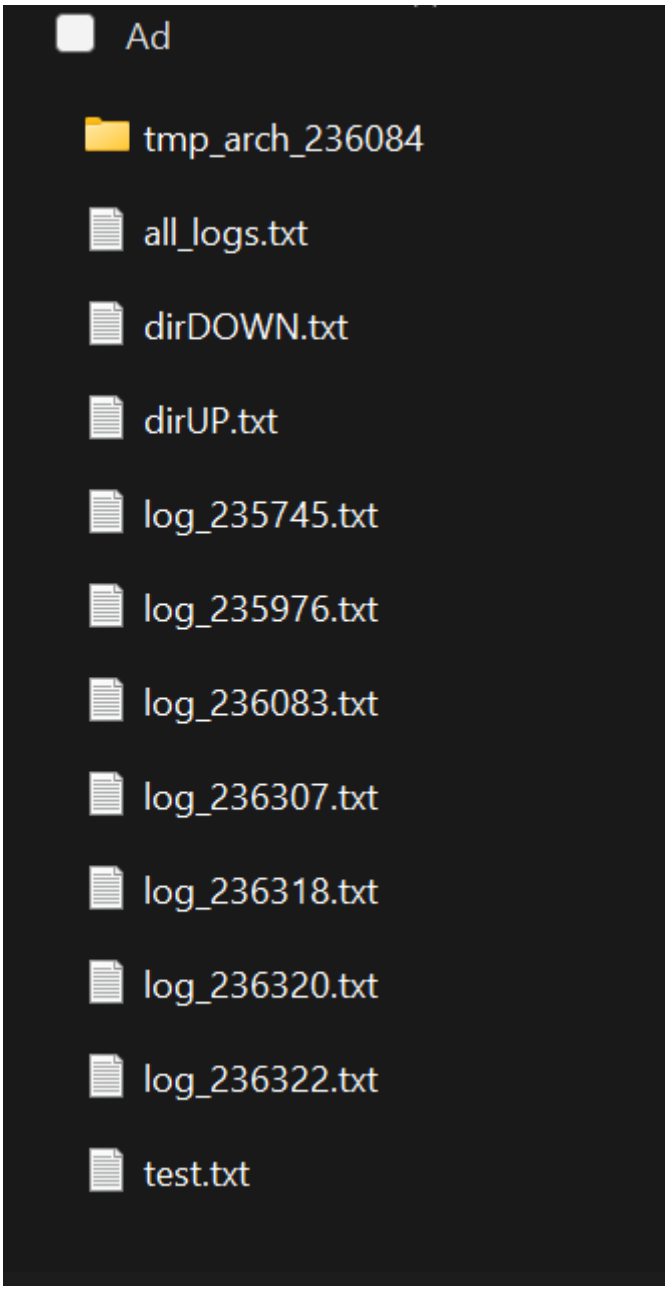
archServer:

```
>>Enter command: archServer archDir
Server response:
Successfully archived files in archDir. Total size: 30720 bytes. Child PID
 237220
>>Enter command:
```

Result:



Contents of tar file:

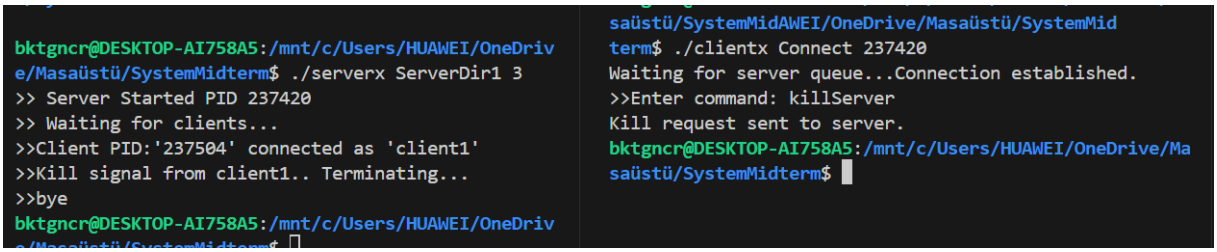


Missing part about tar file: tar file moving into server directorry...

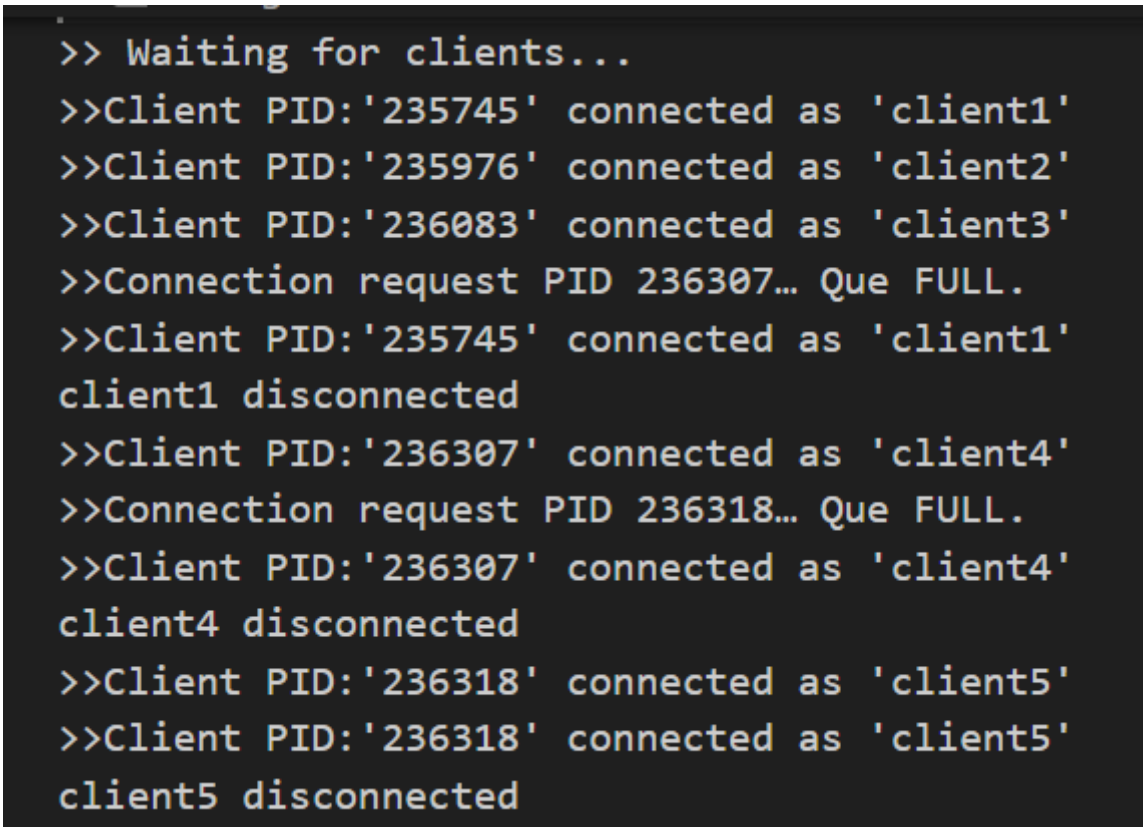
Quit:

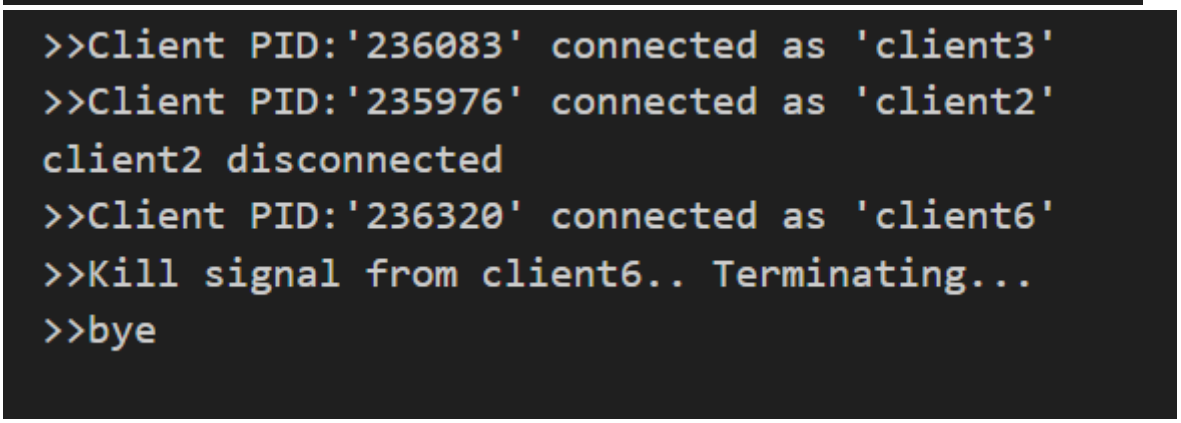
```
>>Enter command: quit
quit request sent to server.
Log message from server: client is shutting down.
```

killServer:


bktgncr@DESKTOP-AI758A5:/mnt/c/Users/HUAWEI/OneDriv
e/Masaüstü/SystemMidterm$ ./serverx ServerDir1 3
>> Server Started PID 237420
>> Waiting for clients...
>>Client PID:'237504' connected as 'client1'
>>Kill signal from client1.. Terminating...
>>bye
bktgncr@DESKTOP-AI758A5:/mnt/c/Users/HUAWEI/OneDriv
saüstü/SystemMidAWEI/OneDrive/Masaüstü/SystemMid
term$ ./clientx Connect 237420
Waiting for server queue...Connection established.
>>Enter command: killServer
Kill request sent to server.
bktgncr@DESKTOP-AI758A5:/mnt/c/Users/HUAWEI/OneDrive/Ma
saüstü/SystemMidterm$

Log file:


>> Waiting for clients...
>>Client PID:'235745' connected as 'client1'
>>Client PID:'235976' connected as 'client2'
>>Client PID:'236083' connected as 'client3'
>>Connection request PID 236307… Que FULL.
>>Client PID:'235745' connected as 'client1'
client1 disconnected
>>Client PID:'236307' connected as 'client4'
>>Connection request PID 236318… Que FULL.
>>Client PID:'236307' connected as 'client4'
client4 disconnected
>>Client PID:'236318' connected as 'client5'
>>Client PID:'236318' connected as 'client5'
client5 disconnected


>>Client PID:'236083' connected as 'client3'
>>Client PID:'235976' connected as 'client2'
client2 disconnected
>>Client PID:'236320' connected as 'client6'
>>Kill signal from client6.. Terminating...
>>bye