



GTU Department of Computer Engineering
CSE 344 - Spring 2024
Midterm Project Report

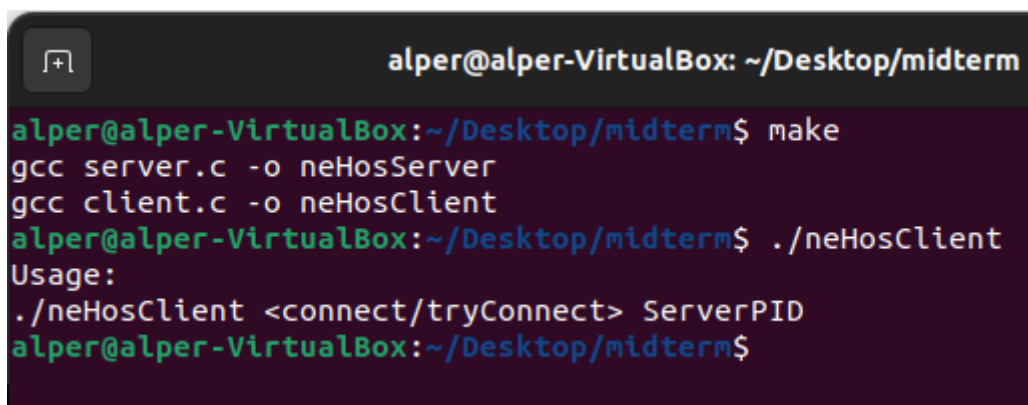
ALPER TAVŞANOĞLU
210104004142

A Concurrent File Access System

In this project, I implement a file server that enables multiple clients to connect, access, modify and archive the files in a specific directory located at the server side. The communication between client and server is handled via FIFOs. The system consists of two main codes: client.c for client-side functions and server.c for server-side operations. The system uses a client-server model where multiple clients can connect to a server to perform file operations like read, write, upload, download, and archive. Inter-process communication (IPC) is achieved through FIFOs, with semaphore and shared memory used for synchronization and mutual exclusion. Critical region handling is done by semaphores.

In client.c:

Client makes a connection request with <connect/tryConnect>. If the queue is not full, Client can connect with both command until que is full. If the queue is full and Client try to connect with “tryConnect” command, then it directly terminates without connecting to the server. On other hand if Client try to connect with “connect” option, then it waits for the empty slot. Waiting process is done by a semaphore created for the client, communicates with a server via FIFOs. And handling Unix signals.

A terminal window with a dark background and light-colored text. The title bar at the top reads 'alper@alper-VirtualBox: ~/Desktop/midterm'. The terminal shows the following commands and output:

```
alper@alper-VirtualBox:~/Desktop/midterm$ make
gcc server.c -o neHosServer
gcc client.c -o neHosClient
alper@alper-VirtualBox:~/Desktop/midterm$ ./neHosClient
Usage:
./neHosClient <connect/tryConnect> ServerPID
alper@alper-VirtualBox:~/Desktop/midterm$
```

Signal Handling Function: sig_handler(int signum)

This function manages signal handling for SIGINT and SIGTERM. It sets a global flag to 1, which is used throughout the program to handle shutdown on signal reception.

Error Handling Function: errExit(char *errorMessage)

A helper function to output errors and exit the program. It simplifies error handling throughout the client code. Exits the program with a failure status to indicate an error condition.

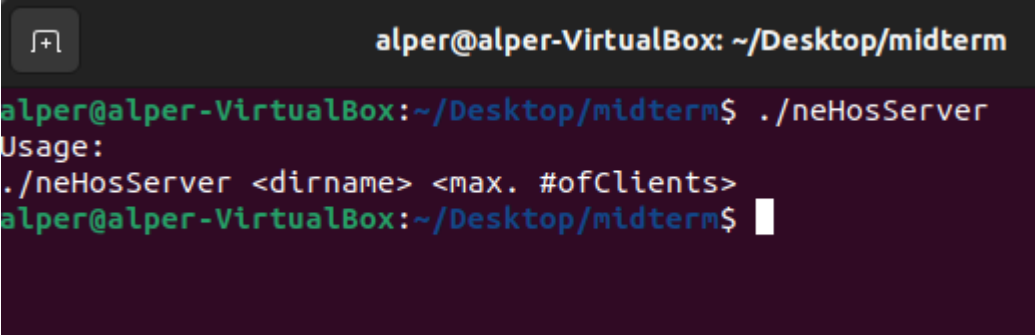
Main Client Function: `client_func(const char sv_no[], int cond)`

Handles the connection operations. It sets up FIFO for communication, writes connection requests, and enters a loop to handle user commands. Manages connection attempts and re-attempts based on server capacity. Processes commands entered by the user and communicates these to the server, handling responses.

And main function: Sets up signal handling and invokes `client_func` with user parameters. `SIGINT`(Ctrl-C) and `SIGTERM` are handling. When `SIGINT` is caught, it terminates the program using the flag and quit from server and removed from the server's clients.

In `server.c`:

This part designed as part of a concurrent file access system that handles multiple client connections, allowing clients to execute file operations in a controlled and synchronized environment. The server-side application of the concurrent file access system manages client requests using multiple processes and Inter-Process Communication (IPC) techniques. It uses FIFOs for messaging, semaphores for synchronization, and shared memory for managing shared data structures.



```
alper@alper-VirtualBox: ~/Desktop/midterm
alper@alper-VirtualBox:~/Desktop/midterm$ ./neHosServer
Usage:
./neHosServer <dirname> <max. #ofClients>
alper@alper-VirtualBox:~/Desktop/midterm$
```

Request Processing Functionality: `void request(char miss_comment[], char user_in[])`

Processes and executes commands received from clients based on the parsed input. The core function that executes the server's responses to client requests, parses the command and executes corresponding operations such as listing files, reading or writing files, and handling uploads/downloads.

File Operation Helpers (`helper_upload`, `helper_download`, `helper_archive`)

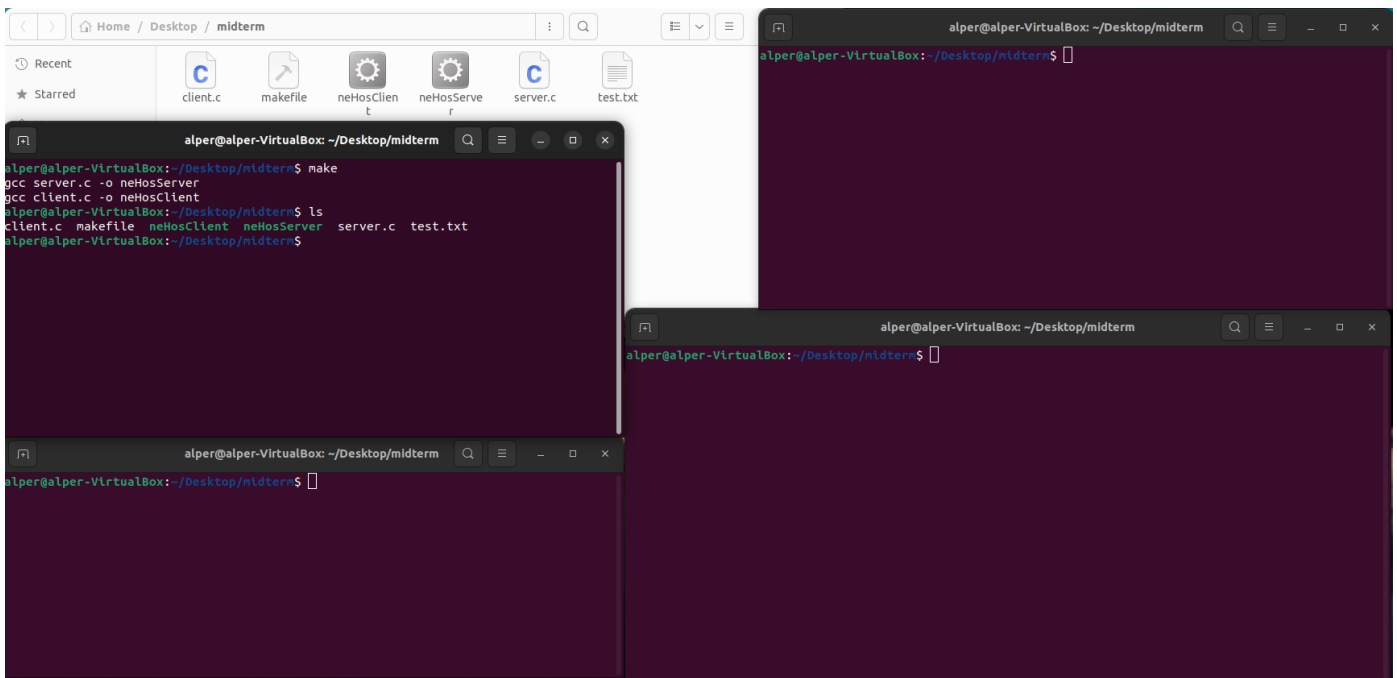
Facilitates critical file operations between the server and clients, ensuring data integrity during transfers. In `helper_archive`, Archives the server's current contents into a specified tar file. Forks a process to execute the tar command using `system`. Checks if the child process exits successfully.

Main Client Function: server_func()

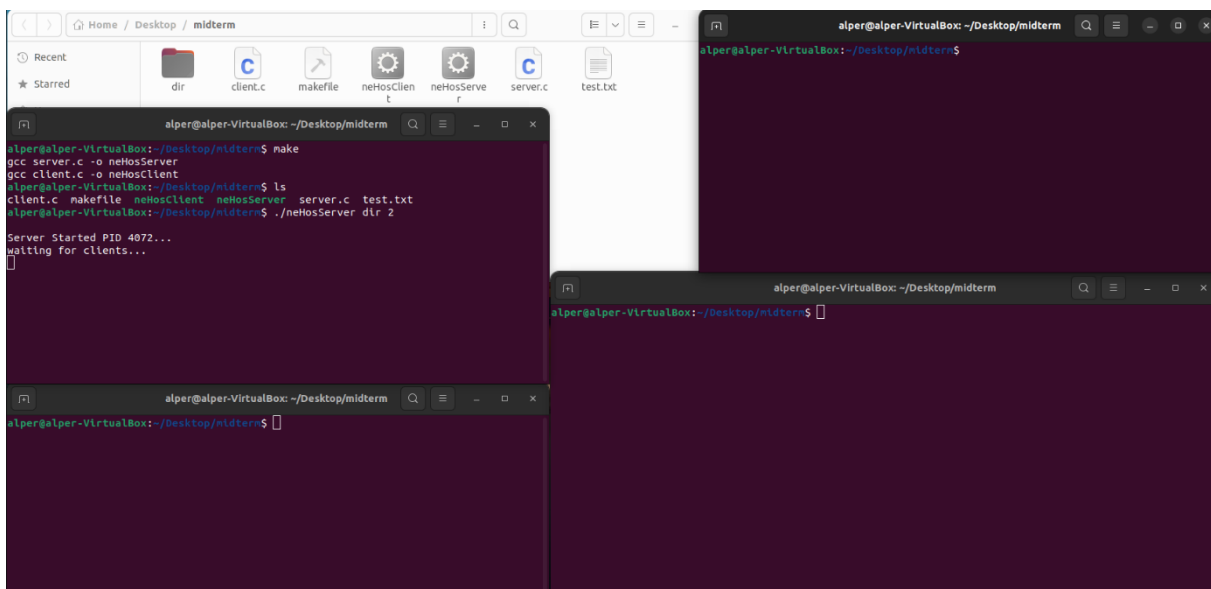
Manages server operations including client connections, request processing, and synchronization. Handles incoming client requests, forks processes for handling each request, manages semaphores for resource access, and logs client activities.

And main function: Entry point of the server program, sets up signal handling, initializes the server environment, and starts server operations. Sets up signal handlers, checks command-line arguments, prepares the server directory and log file, and calls server_func to start the server.

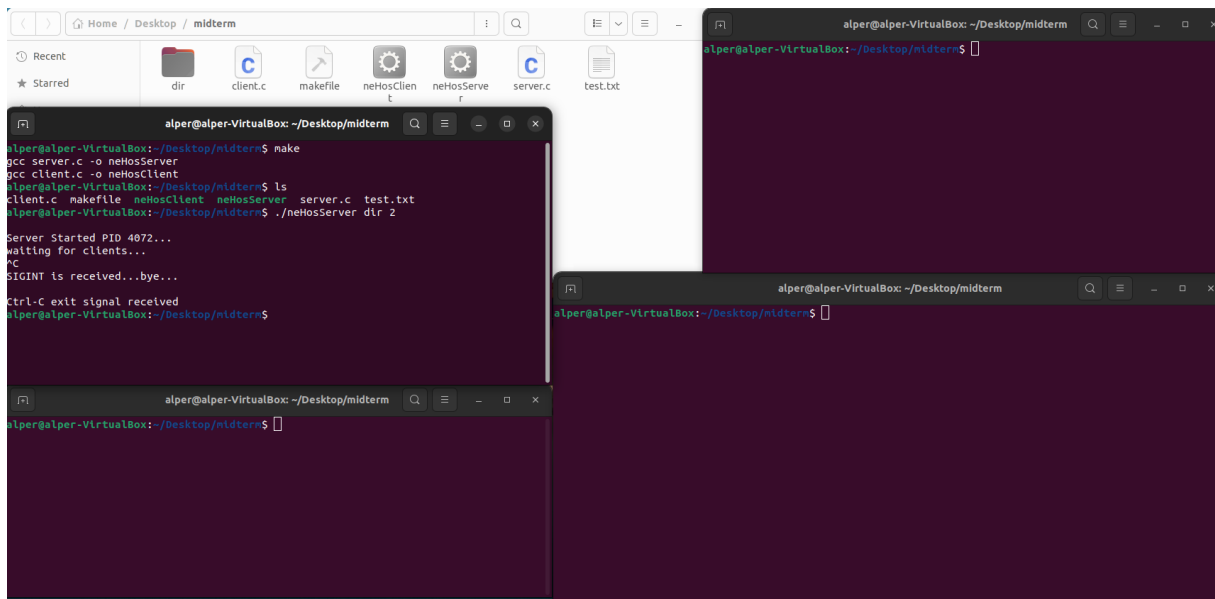
Test Cases and Results



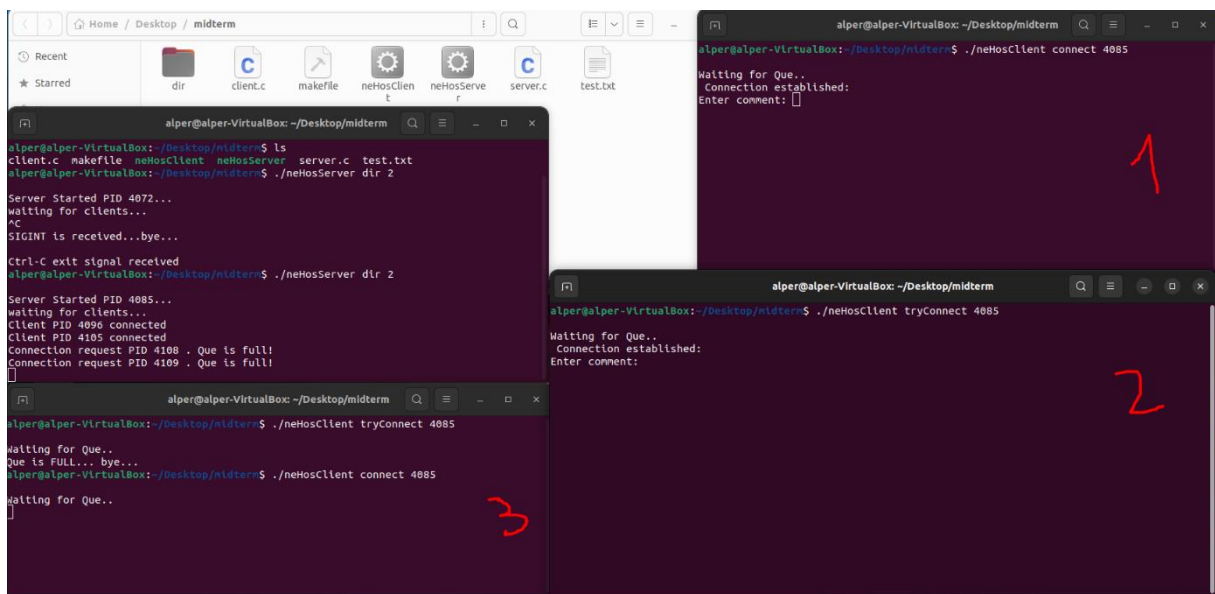
When i run meHosServer with directory name and number of clients, it is creating new file directory.



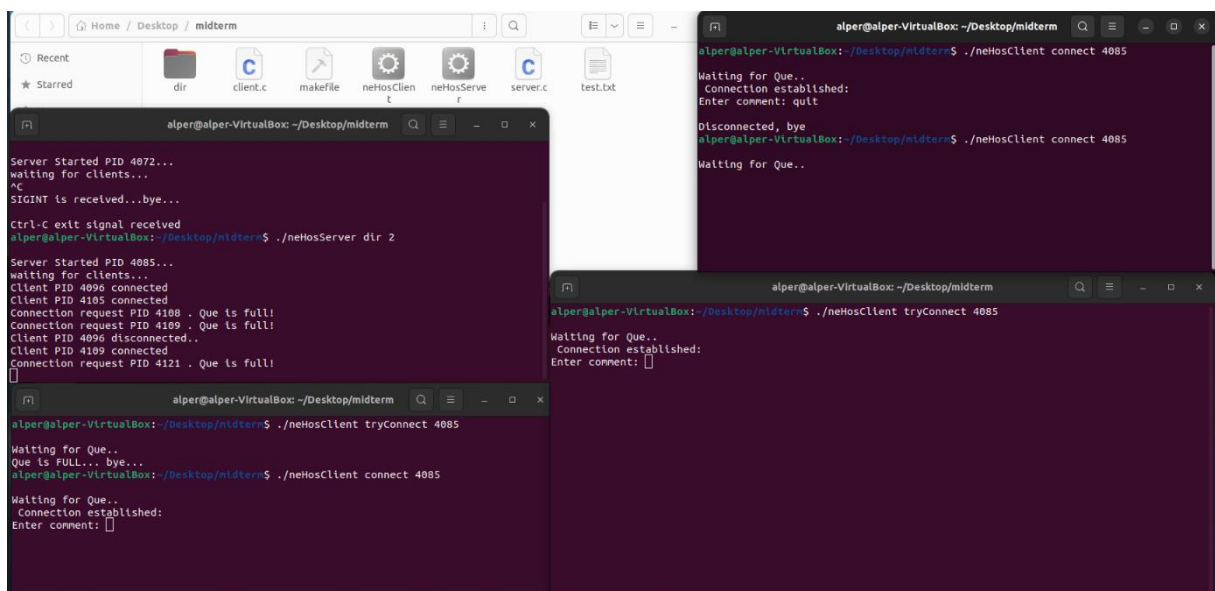
SIGINT/SIGTERM and Connection/Que in Server



The top part of the image shows a file manager window with files: dir, client.c, makefile, neHosClient, neHosServer, server.c, and test.txt. Below it are three terminal windows. The leftmost terminal shows the compilation of server.c and client.c into neHosServer and neHosClient, followed by running neHosServer dir 2. The middle terminal shows the server output: "Server Started PID 4072...", "waiting for clients...", "SIGINT is received...bye...", and "Ctrl-C exit signal received". The rightmost terminal shows the client prompt: "alper@alper-VirtualBox: ~/Desktop/midterm\$".



The middle part of the image shows three terminal windows. The leftmost terminal shows the server output: "Server Started PID 4072...", "waiting for clients...", "SIGINT is received...bye...", "Ctrl-C exit signal received", and then "Server Started PID 4085...", "waiting for clients...", "Client PID 4096 connected", "Client PID 4105 connected", "Connection request PID 4108 . Que is full!", and "Connection request PID 4109 . Que is full!". The middle terminal shows the client running "neHosClient tryConnect 4085", which results in "Waiting for Que..", "Que is FULL... bye...", and "Waiting for Que..". The rightmost terminal shows the client running "neHosClient connect 4085", which results in "Waiting for Que..", "Connection established:", and "Enter comment:". Red handwritten numbers 1, 2, and 3 are placed next to the client's "connect", "tryConnect", and "connect" commands respectively.



The bottom part of the image shows three terminal windows. The leftmost terminal shows the server output: "Server Started PID 4072...", "waiting for clients...", "SIGINT is received...bye...", "Ctrl-C exit signal received", "Server Started PID 4085...", "waiting for clients...", "Client PID 4096 connected", "Client PID 4105 connected", "Connection request PID 4108 . Que is full!", "Connection request PID 4109 . Que is full!", "Client PID 4096 disconnected..", "Client PID 4109 connected", and "Connection request PID 4121 . Que is full!". The middle terminal shows the client running "neHosClient tryConnect 4085", which results in "Waiting for Que..", "Que is FULL... Bye...", and "Waiting for Que..". The rightmost terminal shows the client running "neHosClient connect 4085", which results in "Waiting for Que..", "Connection established:", "Enter comment:", "Disconnected, bye", and "Waiting for Que..".

<- Disconnected

-> Connected

The screenshot shows a file explorer window at the top with the path `Home / Desktop / midterm`. Below it are three terminal windows. The leftmost terminal shows the server starting with `./neHosServer dir 2`, displaying logs for client connections and disconnections. The middle terminal shows the client connecting with `./neHosClient connect 4085`, displaying 'Waiting for Que..', 'Connection established:', and 'Enter comment:'. The rightmost terminal shows the client disconnecting with `./neHosClient connect 4085`, displaying 'Disconnected, bye'.

-> Connected

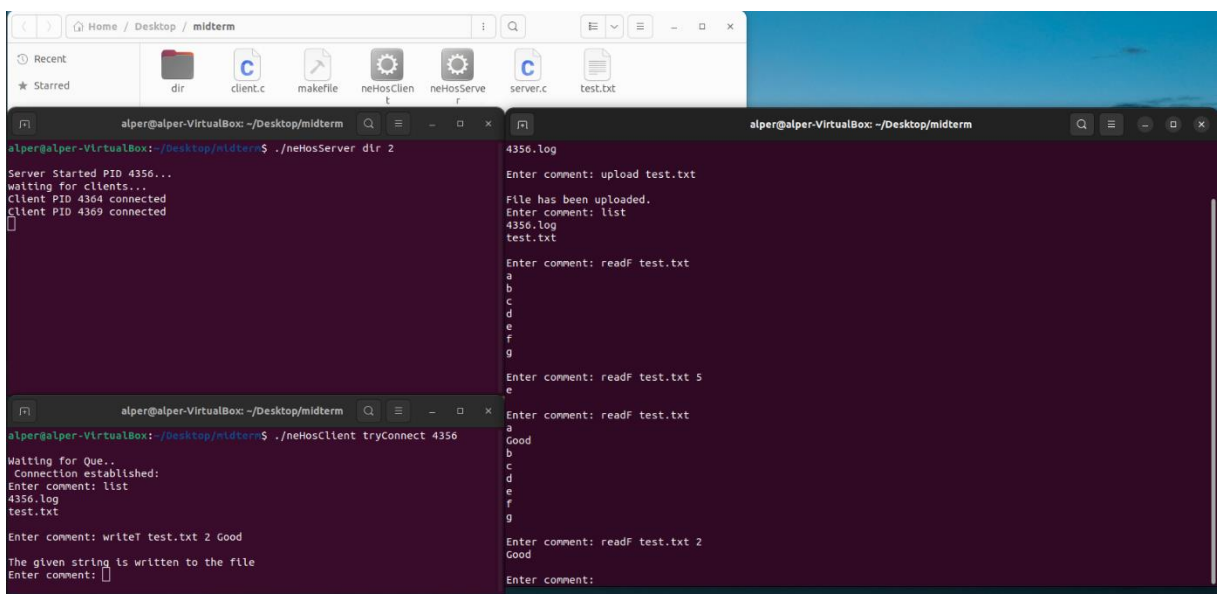
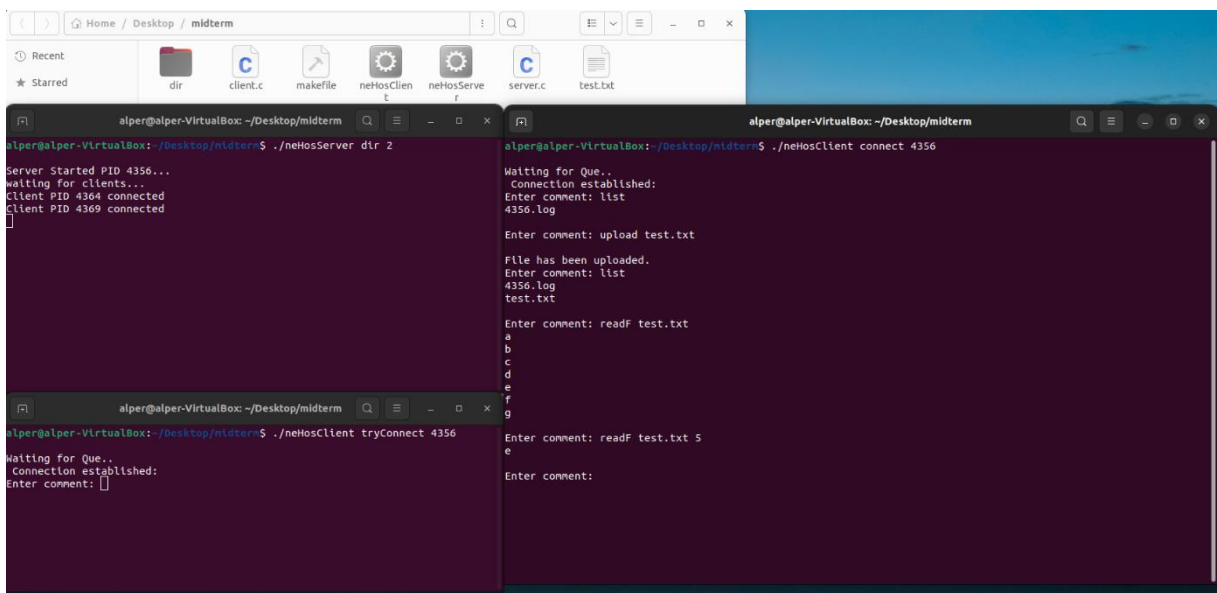
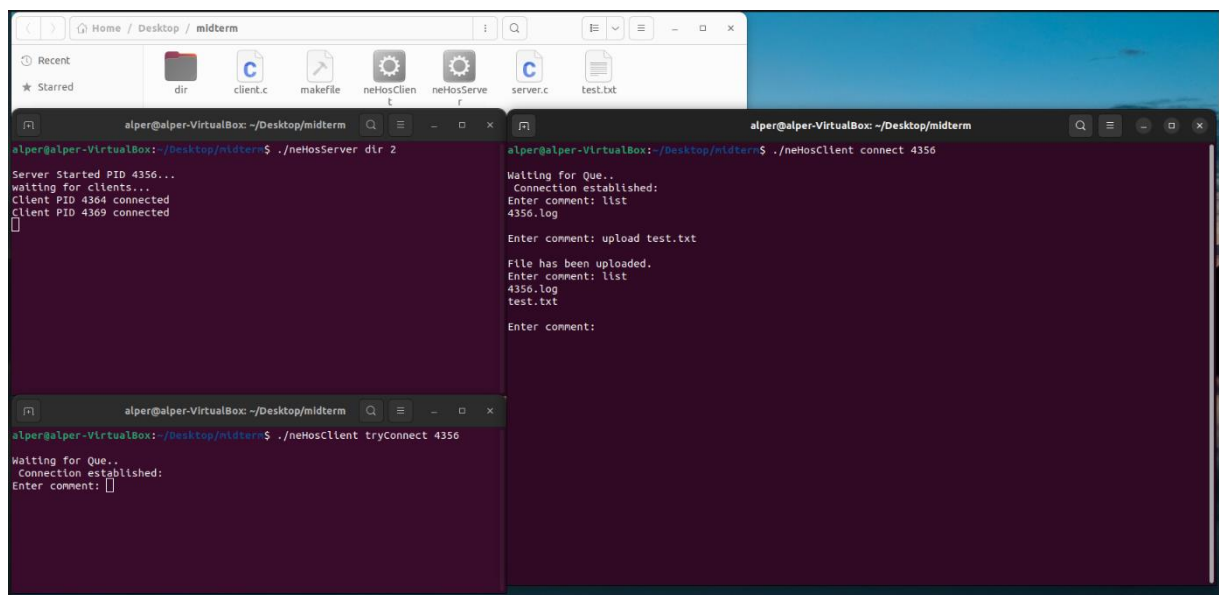
-> Disconnected

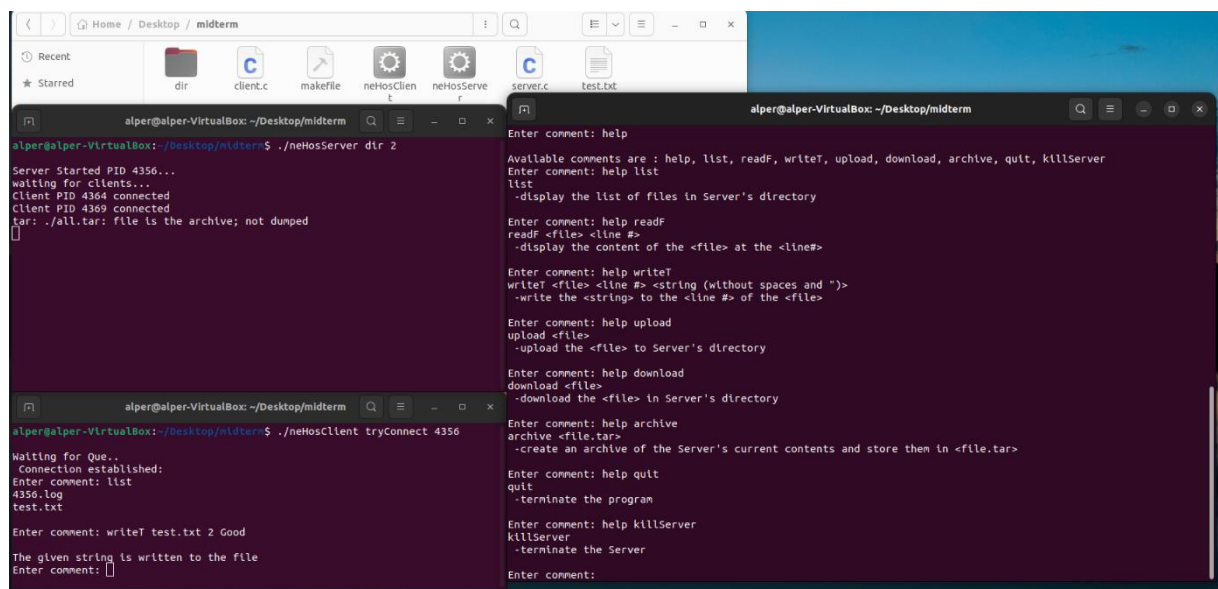
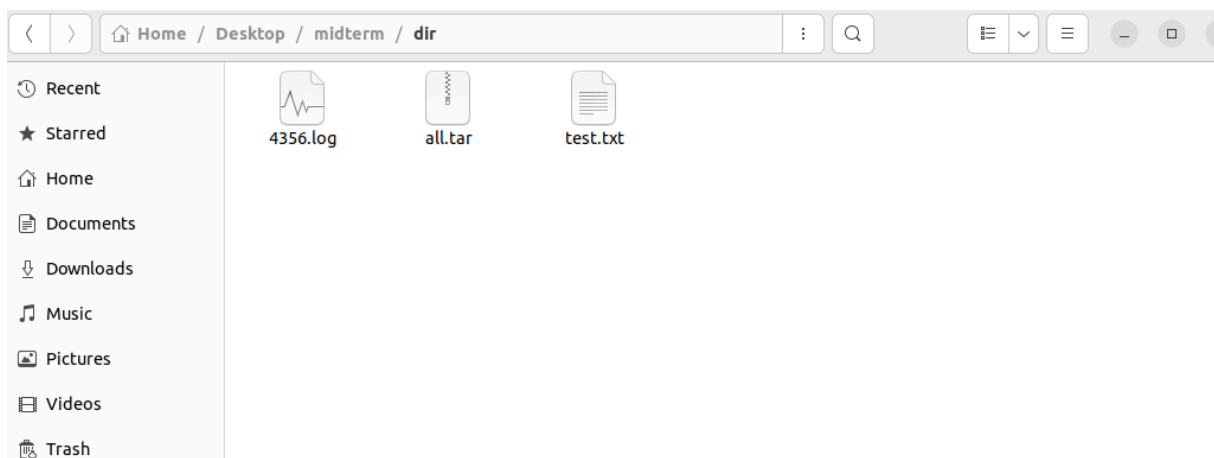
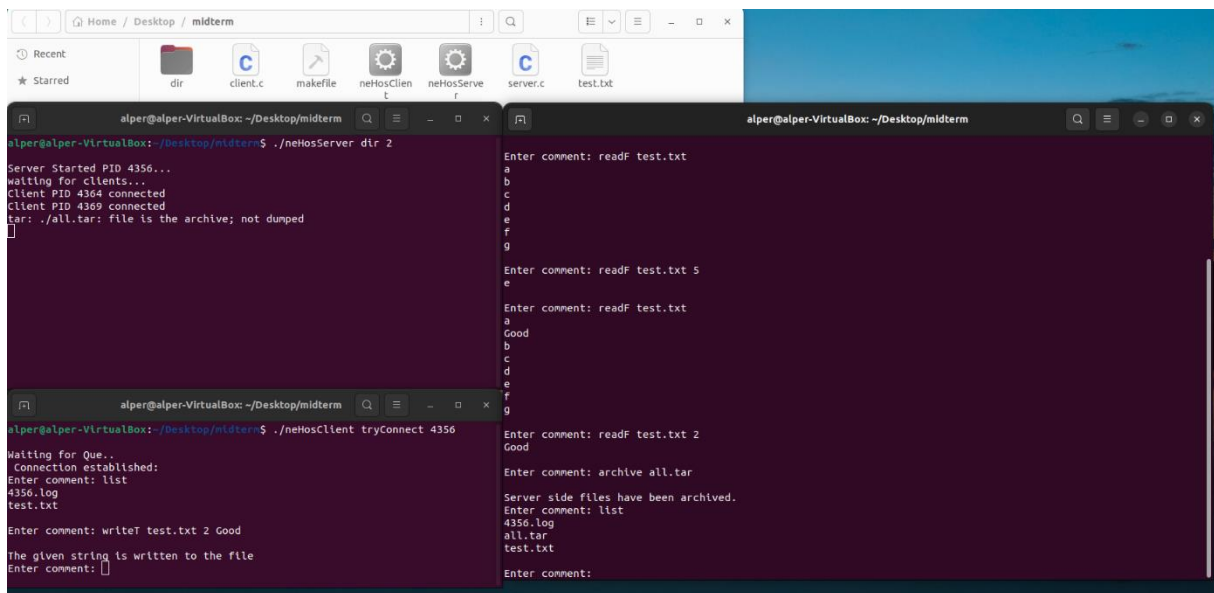
This screenshot shows the same setup as the previous one, but the server has been terminated and restarted. The leftmost terminal shows the server starting again with `./neHosServer dir 2`. The middle terminal shows the client connecting with `./neHosClient connect 4085`. The rightmost terminal shows the client disconnecting with `./neHosClient connect 4085`.

-> Terminated Server

User Commands

This screenshot shows the file explorer and terminal windows with the `killServer` command being used. The leftmost terminal shows the server starting with `./neHosServer dir 2` and then receiving a kill signal from client PID 4170. The middle terminal shows the client connecting with `./neHosClient connect 4156`. The rightmost terminal shows the client disconnecting with `./neHosClient connect 4156`.






```
alper@alper-VirtualBox: ~/Desktop/midterm
alper@alper-VirtualBox: ~/Desktop/midterm$ ./neHosServer dir 2
Server Started PID 4356...
waiting for clients...
Client PID 4364 connected
Client PID 4369 connected
tar: ./all.tar: file is the archive; not dumped
[]

alper@alper-VirtualBox: ~/Desktop/midterm$
Enter comment: writeT test.txt 2 Good
The given string is written to the file
Enter comment: list
4356.log
all.tar
test.txt
Enter comment: download all.tar
File has been downloaded.
Enter comment:
```

```
alper@alper-VirtualBox: ~/Desktop/midterm$
Enter comment: help
Available comments are : help, list, readF, writeF, upload, download, archive, quit, killServer
Enter comment: help list
list
-display the list of files in Server's directory
Enter comment: help readF
readF <file> <line #>
-display the content of the <file> at the <line#>
Enter comment: help writeF
writeF <file> <line #> <string (without spaces and ">
-write the <string> to the <line #> of the <file>
Enter comment: help upload
upload <file>
-upload the <file> to Server's directory
Enter comment: help download
download <file>
-download the <file> in Server's directory
Enter comment: help archive
archive <file.tar>
-create an archive of the Server's current contents and store them in <file.tar>
Enter comment: help quit
quit
-terminate the program
Enter comment: help killServer
killServer
-terminate the Server
Enter comment:
```

```
alper@alper-VirtualBox: ~/Desktop/midterm$ ./neHosServer dir 2
Server Started PID 4356...
waiting for clients...
Client PID 4364 connected
Client PID 4369 connected
tar: ./all.tar: file is the archive; not dumped
Connection request PID 4555 . Que is full!
Connection request PID 4556 . Que is full!
[]

alper@alper-VirtualBox: ~/Desktop/midterm$
Enter comment: writeT test.txt 2 Good
The given string is written to the file
Enter comment: list
4356.log
all.tar
test.txt
Enter comment: download all.tar
File has been downloaded.
Enter comment:
```

```
alper@alper-VirtualBox: ~/Desktop/midterm$ ./neHosClient tryConnect 4356
Waiting for Que..
Que is FULL... bye...
alper@alper-VirtualBox: ~/Desktop/midterm$ ./neHosClient connect 4356
Waiting for Que..
```

```
alper@alper-VirtualBox: ~/Desktop/midterm$
-display the content of the <file> at the <line#>
Enter comment: help writeF
writeF <file> <line #> <string (without spaces and ">
-write the <string> to the <line #> of the <file>
Enter comment: help upload
upload <file>
-upload the <file> to Server's directory
Enter comment: help download
download <file>
-download the <file> in Server's directory
Enter comment: help archive
archive <file.tar>
-create an archive of the Server's current contents and store them in <file.tar>
Enter comment: help quit
quit
-terminate the program
Enter comment: help killServer
killServer
-terminate the Server
Enter comment:
```

```
alper@alper-VirtualBox: ~/Desktop/midterm$ ./neHosServer dir 2
Server Started PID 4356...
waiting for clients...
Client PID 4364 connected
Client PID 4369 connected
tar: ./all.tar: file is the archive; not dumped
Connection request PID 4555 . Que is full!
Connection request PID 4556 . Que is full!
Client PID 4369 disconnected..
Client PID 4556 connected
[]

alper@alper-VirtualBox: ~/Desktop/midterm$
Enter comment: list
4356.log
all.tar
test.txt
Enter comment: download all.tar
File has been downloaded.
Enter comment: ^C
Server Terminated... bye...
Disconnected, bye
alper@alper-VirtualBox: ~/Desktop/midterm$
```

```
alper@alper-VirtualBox: ~/Desktop/midterm$ ./neHosClient tryConnect 4356
Waiting for Que..
Que is FULL... bye...
alper@alper-VirtualBox: ~/Desktop/midterm$ ./neHosClient connect 4356
Waiting for Que..
Connection established:
Enter comment:
```

```
alper@alper-VirtualBox: ~/Desktop/midterm$
-display the content of the <file> at the <line#>
Enter comment: help writeF
writeF <file> <line #> <string (without spaces and ">
-write the <string> to the <line #> of the <file>
Enter comment: help upload
upload <file>
-upload the <file> to Server's directory
Enter comment: help download
download <file>
-download the <file> in Server's directory
Enter comment: help archive
archive <file.tar>
-create an archive of the Server's current contents and store them in <file.tar>
Enter comment: help quit
quit
-terminate the program
Enter comment: help killServer
killServer
-terminate the Server
Enter comment:
```

The screenshot shows a file manager window at the top with files: dir, all.tar, client.c, makefile, neHosClient, neHosServer, server.c, and test.txt. Below are three terminal windows:

- Terminal 1 (Left):** Running `./neHosServer dir 2`. It shows the server starting on PID 4356, waiting for clients, and handling connections from PIDs 4364 and 4369. It also shows a client request for PID 4555 and a disconnection for PID 4369.
- Terminal 2 (Top Right):** Running `./neHosClient connect 4356`. It shows the client connecting to the server, listing files (4356.log, all.tar, test.txt), and reading the content of test.txt.
- Terminal 3 (Bottom Right):** A help menu showing commands: `upload <file>` (upload to server), `download <file>` (download from server), `archive <file.tar>` (create archive), `quit` (terminate program), `killServer` (terminate server), and `help`.

This screenshot shows the next steps in the application:

- Terminal 1 (Left):** The server receives a request from PID 4624. The client (PID 4556) sends the command `download all.tar`, and the server responds with `File has been downloaded.`
- Terminal 2 (Top Right):** The client sends the command `readF test.txt` and receives the content: `a Good b c d e f g`.
- Terminal 3 (Bottom Right):** The help menu is visible, showing the `killServer` command.

This screenshot shows the final steps of the application:

- Terminal 1 (Left):** The server receives a signal from PID 4364 to terminate. It prints `From client PID 4364, Kill signal is received...bye...` and then `Disconnected, bye`. A red arrow points to this line.
- Terminal 2 (Top Right):** The client sends `Server Terminated... bye...` and then `alpergalper-VirtualBox:~/Desktop/midterm$`.
- Terminal 3 (Bottom Right):** The client sends the `killServer` command, and the server responds with `Server is terminated, bye`. A red arrow points to this line.