Capgemini

FTP Server Functional Requirements

Group-3 (Sprint 2)

Customize File Transfer Protocol

Table of content

Chapter 1: Introduction.

The introduction of the software requirement specification provides an overview of the entire Software. The entire SRS with overview description purpose, scope, tools used and basic description. The aim of this document is to gather, analyse and give an in-depth insight into the FTP Server Functional Requirements by defining the problem statement in detail. The detailed requirements of the application are provided in this document.

1.1 Intended Audience

This document is intended to be read by the User.

1.2 Project Purpose:

The purpose of this project is to build and server client application which handles multiple clients. Where authorised clients can browse, update and download the files from server and anonymous clients can only upload the file to server.

1.3 Key project Objectives:

a. Server initialization message is displayed after starting server
b. Client connects message is displayed after clients connects to server
c. Authenticated and Anonymous user logins
d. Multiple clients connected to a single server
e. Client will have multiple options to browse, upload and download files from server
f. Clients show the successful and unsuccessful message after performing a particular operation

1.4 Project Scope:

The main aim of the project is to login a authenticated and anonymous users and do the required operations such as browse, list, upload, download and view present directory of the user, where as the anonymous user will not be having download privilege and whereas all the operations performed by anonymous user will only be effective in public directory.

2. Design Overview: -

- **Customize File transfer Protocols comprises of the following function in maintain database:**

Name of the Module	Browse choice
Handled by	
Description	The function is used to list the contents of the directory.

Name of the Module	Read choice
Handled by	
Description	This function is used to print the contents inside the file.

Name of the Module	PWD command
Handled by	
Description	The function is used to know present working directory

Name of the Module	Upload
Handled by	
Description	This function is used to upload file on the server

Name of the Module	Download
Handled by	
Description	This function is used to download file from server

Name of the Module	Bye command
Handled by	
Description	This command is used to exit the client from the server.

1. Design Objectives:

1. Start the connection
2. Accept the connection
3. Check for black list IP
4. Different choices should be able to get excepted output.
5. Customize file transfer protocol able to send file.

2. Design Alternative: -

We have used fork instead of thread for independent multiple client handling.

2.3 User Interface Paradigms: -

The Customize file transfer protocol should be able to allow the client to upload, download, read the contents file and browse directory. Multiple clients should be able to connect one server.

1. Validation: -

- The server first check for IP of incoming client check in black list IP file accordingly connects to client
- The server and Client should be able to establish a connection successfully.

- There are two types of users Anonymous and Authenticated User. For anonymous users ’username is any value other than valid username. But for authenticated users username and password credentials should match with the valid_user.txt file.

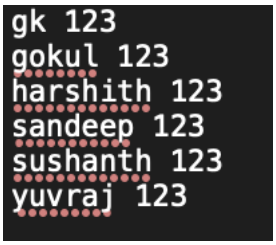
- Authenticate function load the user details from the valid_user.txt file give the user access accordingly.

3. SYSTEM ARCHITECTURE: -

3.1. Database Architecture

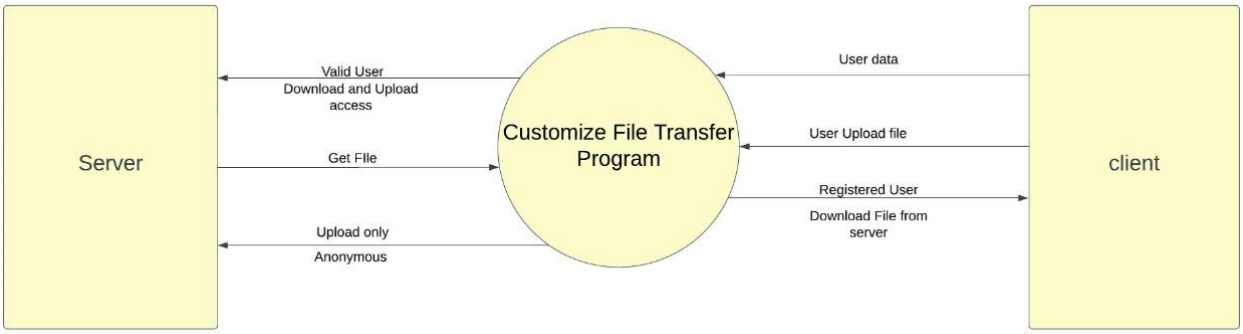
The database used inside our program is the user.txt file. The user.txt file contains username and password separated by space. We have used strtok with a delimiter as space to store username and password. Also, for the present directory we can use username for the purpose.

The architecture used in this system is like that we store all the users inside a data folder in the home directory. And all the information of the authenticated and anonymous user stored in user.txt files.

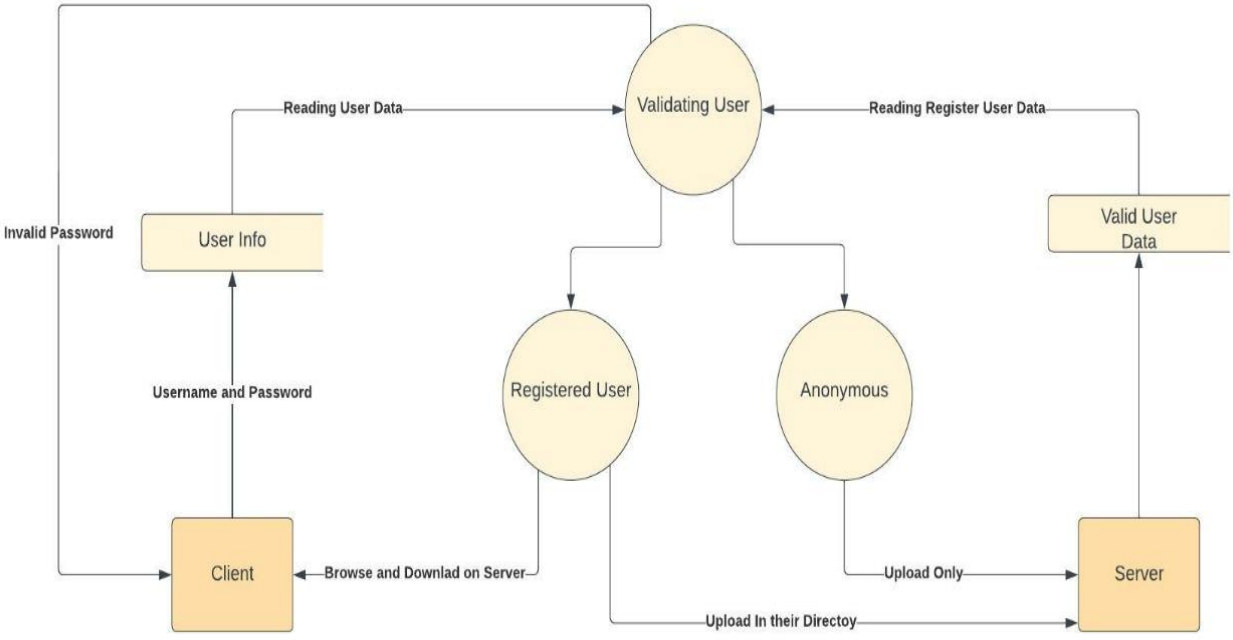


4. Detailed System Design:

4.1 DFD-0:



4.2: DFD-1:



5. Tools report

5.1 Valgrind tool

```
==16835== Memcheck, a memory error detector
==16835== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==16835== Using Valgrind-3.16.1 and LibVEX; rerun with -h for copyright info
==16835== Command: ./a.out
==16835==
[+]Server socket created successfully.
[+]Binding successfull.
[+]Listening....
^C==16835==
==16835== Process terminating with default action of signal 2 (SIGINT)
==16835==    at 0x4949AB3: accept (accept.c:26)
==16835==    by 0x10B498: main (in /home/cg83-user20/CGSprint2/capg2/capg/CUT/Code/Server/src/a.out)
==16835==
==16835== HEAP SUMMARY:
==16835==     in use at exit: 0 bytes in 0 blocks
==16835==   total heap usage: 1 allocs, 1 frees, 1,024 bytes allocated
==16835==
==16835== All heap blocks were freed -- no leaks are possible
==16835==
==16835== For lists of detected and suppressed errors, rerun with: -s
==16835== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

Server report

```
==16892== Memcheck, a memory error detector
==16892== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==16892== Using Valgrind-3.16.1 and LibVEX; rerun with -h for copyright info
==16892== Command: ./a.out
==16892==
[+]Server socket created successfully.
[-]Error in socket: Connection refused
==16892==
==16892== HEAP SUMMARY:
==16892==     in use at exit: 0 bytes in 0 blocks
==16892==   total heap usage: 5 allocs, 5 frees, 3,064 bytes allocated
==16892==
==16892== All heap blocks were freed -- no leaks are possible
==16892==
==16892== For lists of detected and suppressed errors, rerun with: -s
==16892== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```

Client report:

5.2 Gcov report

Server Report for Authenticated user:

```
File 'client.c'
Lines executed:84.72% of 144
Creating 'client.c.gcov'
```

Client report for authenticated user:

```
File 'server.c'
Lines executed:77.78% of 153
Creating 'server.c.gcov'
```

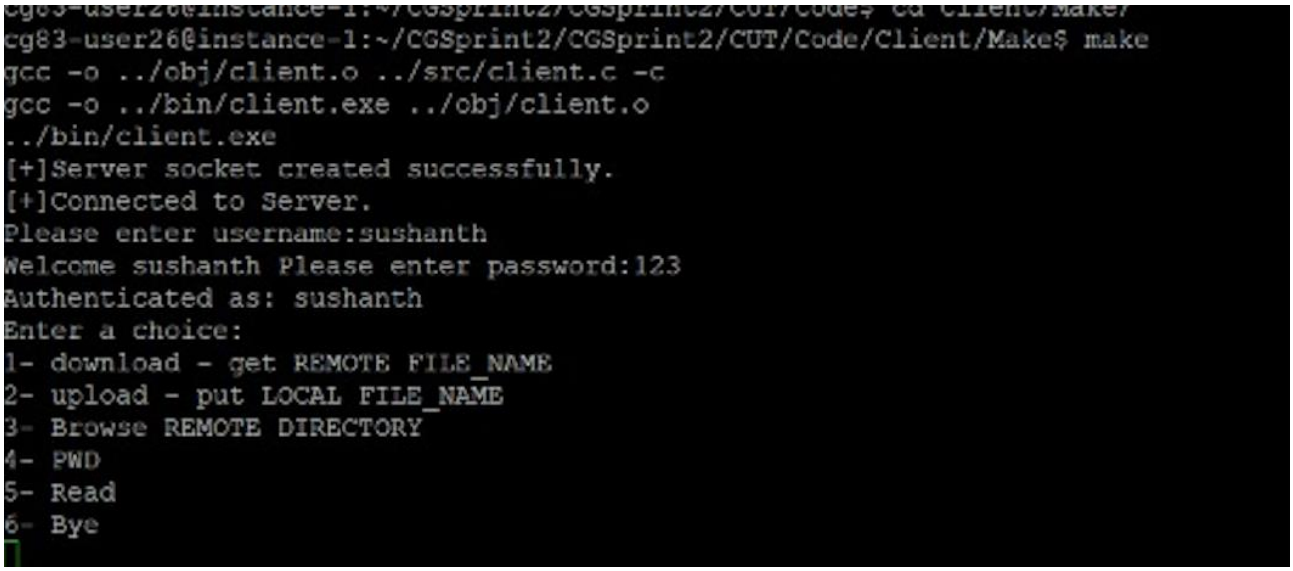
6. Testing

6.1 Unit Testing(CUnit):

```
    CUnit - A unit testing framework for C - Version 2.1-3
    http://cunit.sourceforge.net/


Suite: Testing_Suite1
  Test: Testing Sunny Cases ...


passed
  Test: Testing Rainy Cases ...


passed
  Test: Testing Sunny of authenticate Cases ...passed
  Test: Testing Rainy of authenticate Cases ...passed

Run Summary:    Type  Total    Ran Passed Failed Inactive
              suites      1      1    n/a      0        0
               tests      4      4      4      0        0
             asserts     12     12     12      0      n/a

Elapsed time =    0.000 seconds
```

6.2 Integration Testing:

6.2.1: Authenticated User Login



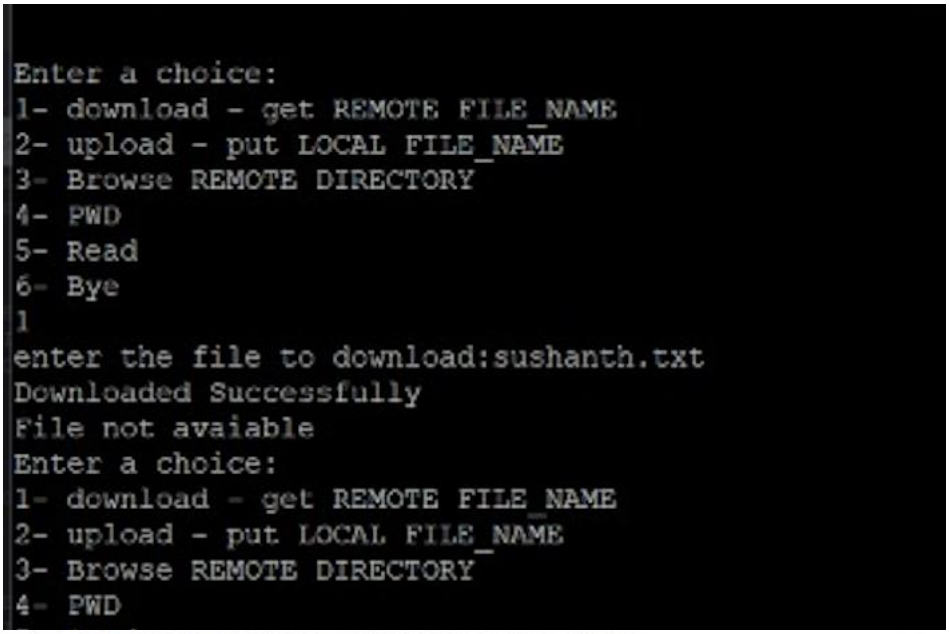
6.2.2: Authenticated User pwd:

6.2.3: ls command

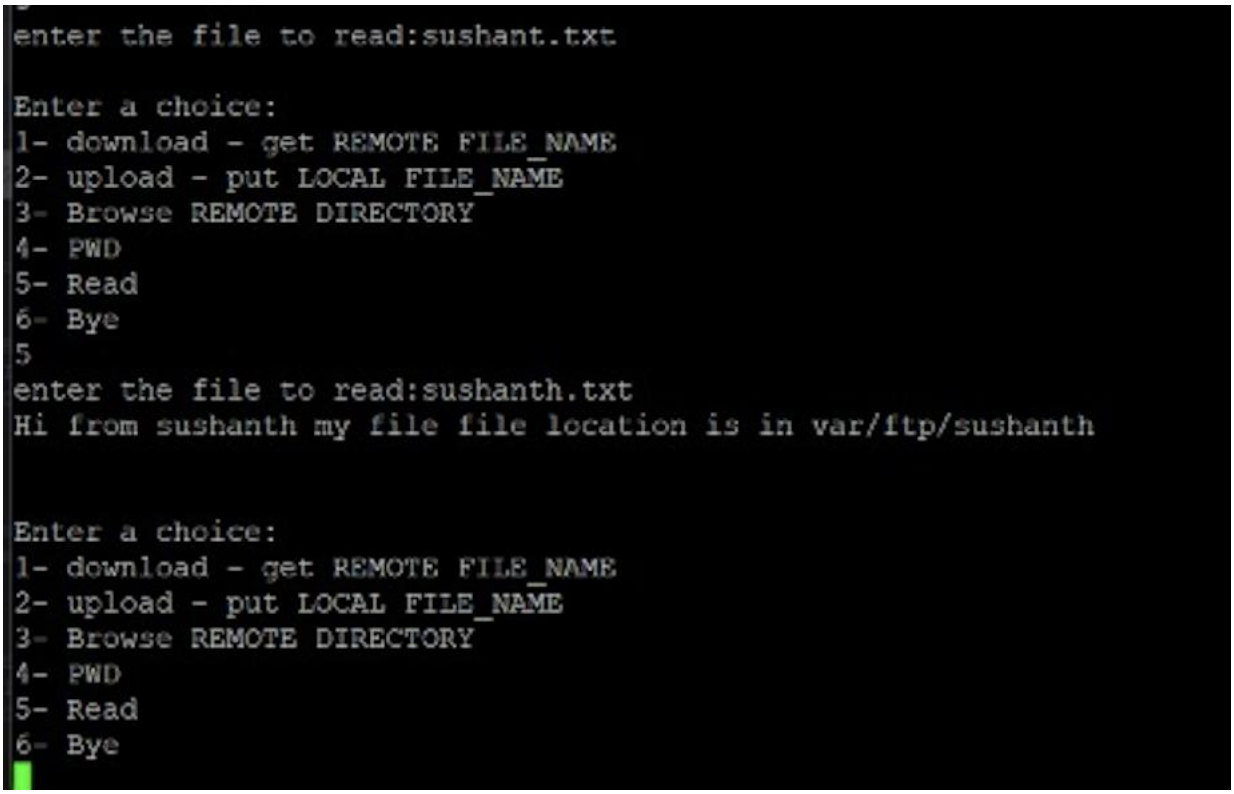
```
Please enter username:sushanth
Welcome sushanth Please enter password:123
Authenticated as: sushanth
Enter a choice:
1- download - get REMOTE FILE_NAME
2- upload - put LOCAL FILE_NAME
3- Browse REMOTE DIRECTORY
4- PWD
5- Read
6- Bye
3
cli.txt
sushanth.txt

Enter a choice:
1- download - get REMOTE FILE_NAME
2- upload - put LOCAL FILE_NAME
3- Browse REMOTE DIRECTORY
4- PWD
5- Read
6- Bye
```

6.2.4: Download File from server:

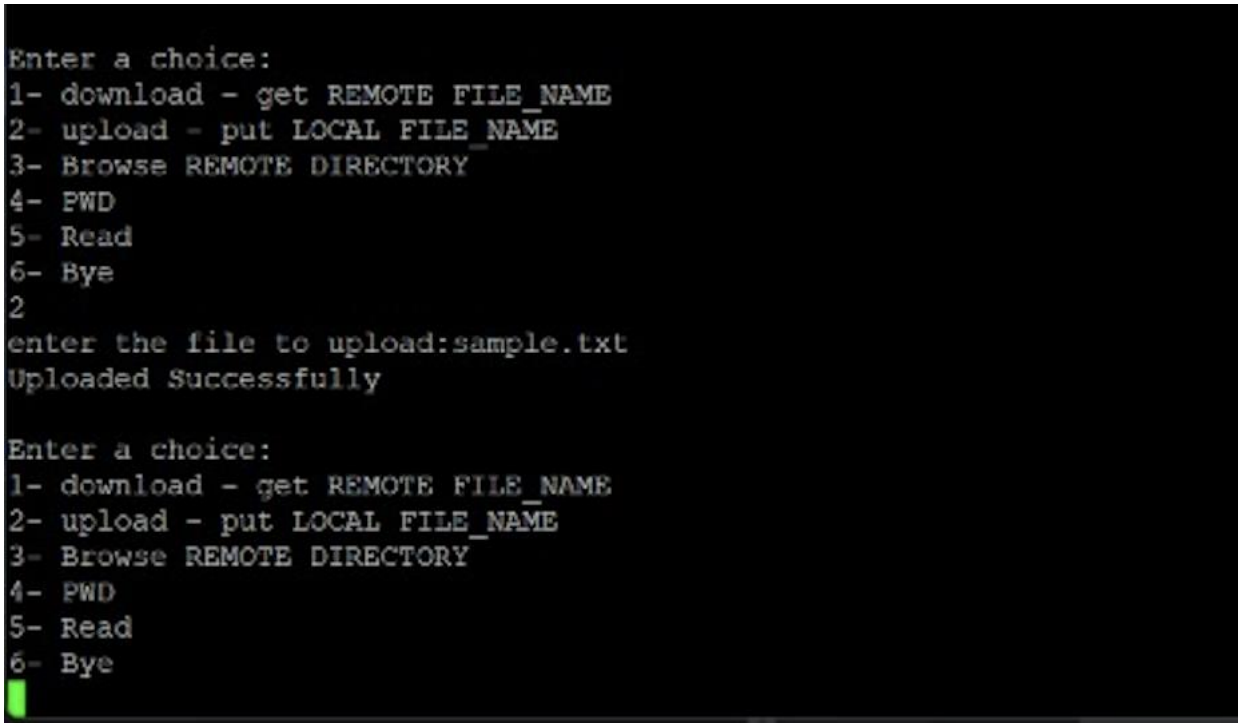
```
Enter a choice:
1- download - get REMOTE FILE_NAME
2- upload - put LOCAL FILE_NAME
3- Browse REMOTE DIRECTORY
4- PWD
5- Read
6- Bye
4
/home/cg83-user26/CGSprint2/CGSprint2/CUT/Code/Server/var/ftp/sushanth

Enter a choice:
1- download - get REMOTE FILE_NAME
2- upload - put LOCAL FILE_NAME
3- Browse REMOTE DIRECTORY
4- PWD
5- Read
6- Bye
```



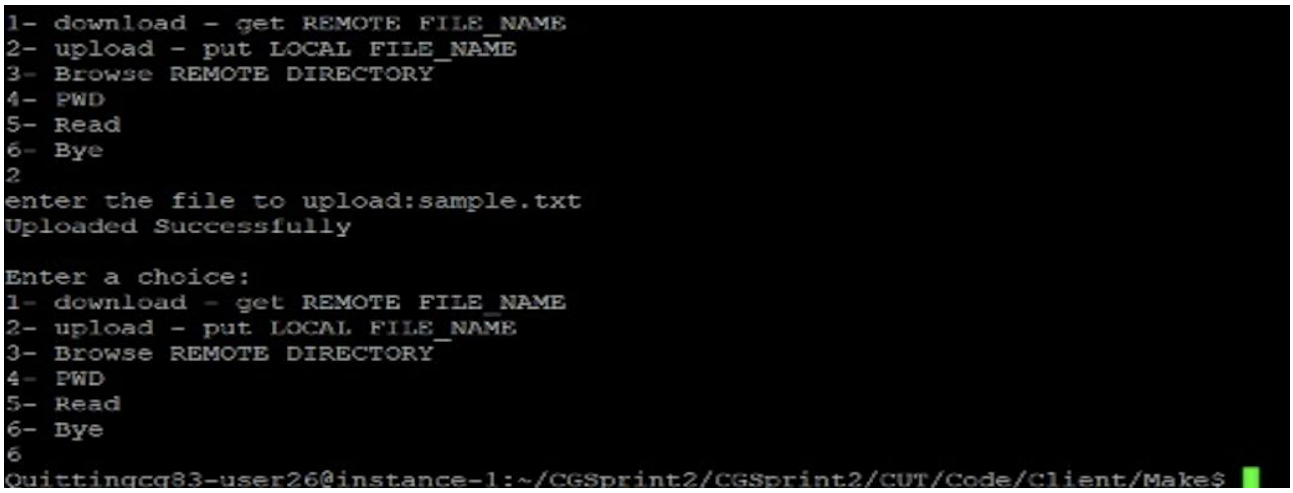
6.2.5: Cat Command



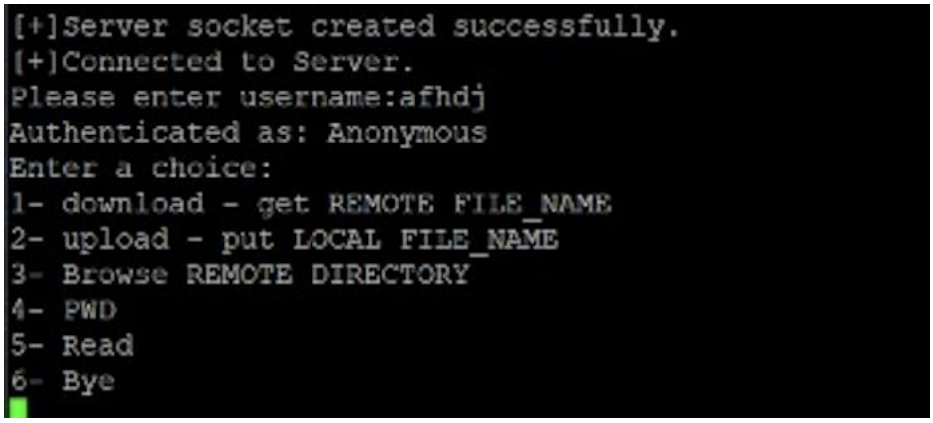
6.2.6: Uplaod file from client to users directory on server



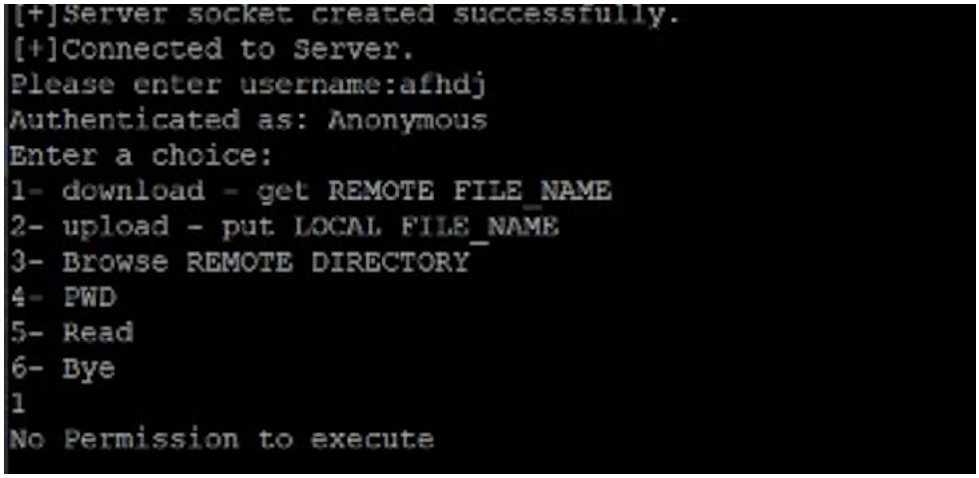
6.2.7: Quitting Server



6.2.8: Anonymous user login



6.2.9: Restricting download privilege for anonymous user



6.2.10: directing anonymous user to public directory

7. Requirements Traceability Matrix(RTM)

Req	Design Ma	Code Mapping	UT Mapping	IT Mapping
RLET_01	3.1.1	Authenticate user login	Unit Test Case7_9	IT_CASE 1
RLET_2	3.1.2	Authenticated user pwd		IT_CASE 2
RLET_3	3.1.3	ls command		IT_CASE 3
RLET_4	3.1.4	Download file from server		IT_CASE 4
RLET_5	3.1.5	cat command		IT_CASE 5
RLET_6	3.1.6	uplaod file form client to user directory on server		IT_CASE 6
RLET_7	3.1.7	Quitting server		IT_CASE 7
RLET_8	3.1.8	Anonymous user login	Unit Test Case10_12	IT_CASE 8
RLET_9	3.1.9	Restricting download privilege		IT_CASE 9
RLET_10	3.1.10	directiong anonymous user to public directory		IT_CASE 10
RLET_11	3.1.11	check blacklist	Unit Test Case1_6	IT_CASE 11