Project 1 - Implementing telnet client and server with Python

Introduction

In our project we have two files, one on the server side named **stelnet.py** and other on the client side named **ctelnet.py**. We used socket programming and TCP protocol to establish client server connection and perform telnet commands. The client enters a command, we code the server side to handle the command, server generates the response, then server processes the command and gives the response back to client for every command.

Establishment of Client-Server Communication

In order to run our telnet commands, we firstly need to establish the connection between client and server. Once the connection is established, we can run the telnet commands. **127.0.0.1** is the **IP address** and **10100** is the **port number** of the server.

Steps to run:

Open a terminal on Virtual Machine and type: **python stelnet.py**
Open another terminal on Virtual Machine and type: **python ctelnet.py 127.0.0.1 10100**

The entire implementation is using **Python2**

Establishment of server-side code

```
from socket import *
import ftplib
from ftplib import FTP
import os
import stat
import sys
import shutil
import requests
import subprocess
import httplib2
#from urllib.parse import urlencode

serverName = "localhost"
serverPort = 10100
serverSocket = socket(AF_INET,SOCK_STREAM)
serverSocket.bind(('',serverPort))
serverSocket.listen(1)

ftp = ftplib.FTP(serverName)

ftp.login("seed","dees")

print ('The server is ready to receive')
while 1:
```

FIGURE 1:SERVER-SIDE MAIN CODE

Establishment of client-side code

```
from socket import *
import sys
from ftplib import FTP

clientsys = sys.argv

if(clientsys < 3) :
                print 'Usage.. : python telnet.py hostname port'
                sys.exit()
serverName = clientsys[1] #127.0.0.1
serverPort = int(clientsys[2]) #10100
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName,serverPort))

ftp = FTP(serverName)
ftp.login("seed","dees")

input1 = raw_input("Enter a command:")

clientSocket.send(input1.encode())
modifiedSentence = clientSocket.recv(1024)
outputF = modifiedSentence.decode()
print ('From Server:', outputF)
clientSocket.close()
```

FIGURE 2: CLIENT-SIDE MAIN CODE

Since we have established the connection between the server and the client, we can now run the telnet commands.

Steps to run:

1) **HEAD / HTTP/1.0**

```
elif(input1.startswith("HEAD /HTTP")):
        http = httplib2.Http()
        resp = http.request("http://localhost","HEAD")[0]
        print(resp)
        output = "Response Received"
        connectionSocket.send(output.encode())
```

FIGURE 3: HEAD CODE

```
The server is ready to receive
{'status': '200', 'content-length': '3186', 'content-lo
cation': 'http://localhost', 'content-encoding': 'gzip'
, 'accept-ranges': 'bytes', 'vary': 'Accept-Encoding',
'server': 'Apache/2.4.18 (Ubuntu)', 'last-modified': 'T
ue, 25 Jul 2017 23:45:38 GMT', 'etag': '"2c39-5552cedad
62b8-gzip"', 'date': 'Sat, 29 Feb 2020 01:26:10 GMT', '
content-type': 'text/html'}
```

FIGURE 4:HEAD SERVER-SIDE OUTPUT

```
[02/28/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:HEAD /HTTP/1.0
('From Server:', u'Response Received')
```

FIGURE 5: HEAD CLIENT-SIDE OUTPUT

2) GET / HTTP/1.0

```
elif(input1.startswith("GET /HTTP/")):
        http = httplib2.Http()
        contents = http.request("http://localhost")[1]
        print(contents)
        connectionSocket.send(contents)
```

FIGURE 6:GET CODE

```
The server is ready to receive
b'\n<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transi
tional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-tra
nsitional.dtd">\n<html xmlns="http://www.w3.org/1999/xh
tml">\n  <!--\n    Modified from the Debian original fo
r Ubuntu\n    Last updated: 2014-03-19\n    See: https:
//launchpad.net/bugs/1288690\n  -->\n  <head>\n    <met
a http-equiv="Content-Type" content="text/html; charset
=UTF-8" />\n    <title>Apache2 Ubuntu Default Page: It
works</title>\n    <style type="text/css" media="screen
">\n  * {\n    margin: 0px 0px 0px 0px;\n    padding: 0
px 0px 0px 0px;\n  }\n\n  body, html {\n    padding: 3p
x 3px 3px 3px;\n\n    background-color: #D8DBE2;\n\n
 font-family: Verdana, sans-serif;\n    font-size: 11pt
;\n    text-align: center;\n  }\n\n  div.main_page {\n
   position: relative;\n    display: table;\n\n    widt
h: 800px;\n\n    margin-bottom: 3px;\n    margin-left:
auto;\n    margin-right: auto;\n    padding: 0px 0px 0p
x 0px;\n\n    border-width: 2px;\n    border-color: #21
2738;\n    border-style: solid;\n\n    background-color
```

```
 which provides better security out of the box.\n
      </p>\n        </div>\n\n        <div class="secti
on_header">\n          <div id="bugs"></div>\n
       Reporting Problems\n        </div>\n        <div
 class="content_section_text">\n          <p>\n
        Please use the <tt>ubuntu-bug</tt> tool to repo
rt bugs in the\n                Apache2 package with Ub
untu. However, check <a\n                href="https://
bugs.launchpad.net/ubuntu/+source/apache2">existing\n
              bug reports</a> before reporting a new bu
g.\n          </p>\n          <p>\n                Plea
se report bugs specific to modules (such as PHP and oth
ers)\n              to respective packages, not to th
e web server itself.\n          </p>\n        </div>\n\
n\n\n\n      </div>\n    </div>\n    <div class="valida
tor">\n    </div>\n  </body>\n</html>\n\n'
```

FIGURE 7:SERVER-SIDE OUTPUT

```
input a command: GET /HTTP/1.0

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transition
al//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transit
ional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2014-03-19
    See: https://launchpad.net/bugs/1288690
  -->
  <head>
    <meta http-equiv="Content-Type" content="text/html;
 charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title
>
    <style type="text/css" media="screen">
  * {
    margin: 0px 0px 0px 0px;
    padding: 0px 0px 0px 0px;
    padding: 0px 0px 0px 0px;
  }

  body, html {
    padding: 3px 3px 3px 3px;

    background-color: #D8DBE2;

    font-family: Verdana, sans-serif;
    font-size: 11pt;
    text-align: center;
  }

  div.main_page {
    position: relative;
    display: table;

    width: 800px;

    margin-bottom: 3px;
    margin-left: auto;
    margin-right: auto;
    text-align: center;
  }

  div.main_page {
    position: relative;
    display: table;

    width: 800px;

    margin-bottom: 3px;
    margin-left: auto;
    margin-right: auto;
    padding: 0px 0px 0px 0px;

    border-width: 2px;
    border-color: #212738;
    border-style: solid;

    background-color: #FFFFFF;

    text-al
```

FIGURE 8: GET CLIENT-SIDE OUTPUT

3) GET / HTTP/1.1

```
elif(input1.startswith("GET /HTTP/")):
        http = httplib2.Http()
        contents = http.request("http://localhost")[1]
        print(contents)
        connectionSocket.send(contents)
```

FIGURE 9:GET CODE

```
The server is ready to receive
b'\n<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transi
tional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-tra
nsitional.dtd">\n<html xmlns="http://www.w3.org/1999/xh
tml">\n  <!--\n    Modified from the Debian original fo
r Ubuntu\n    Last updated: 2014-03-19\n    See: https:
//launchpad.net/bugs/1288690\n  -->\n  <head>\n    <met
a http-equiv="Content-Type" content="text/html; charset
=UTF-8" />\n    <title>Apache2 Ubuntu Default Page: It
works</title>\n    <style type="text/css" media="screen
">\n  * {\n    margin: 0px 0px 0px 0px;\n    padding: 0
px 0px 0px 0px;\n  }\n\n  body, html {\n    padding: 3p
x 3px 3px 3px;\n\n    background-color: #D8DBE2;\n\n
 font-family: Verdana, sans-serif;\n    font-size: 11pt
;\n    text-align: center;\n  }\n\n  div.main_page {\n
   position: relative;\n    display: table;\n\n    widt
h: 800px;\n\n    margin-bottom: 3px;\n    margin-left:
auto;\n    margin-right: auto;\n    padding: 0px 0px 0p
x 0px;\n\n    border-width: 2px;\n    border-color: #21
2738;\n    border-style: solid;\n\n    background-color
```

```
 which provides better security out of the box.\n
      </p>\n        </div>\n\n        <div class="secti
on_header">\n          <div id="bugs"></div>\n
      Reporting Problems\n        </div>\n        <div
 class="content_section_text">\n          <p>\n
       Please use the <tt>ubuntu-bug</tt> tool to repo
rt bugs in the\n                Apache2 package with Ub
untu. However, check <a\n                href="https://
bugs.launchpad.net/ubuntu/+source/apache2">existing\n
            bug reports</a> before reporting a new bu
g.\n          </p>\n          <p>\n                Plea
se report bugs specific to modules (such as PHP and oth
ers)\n              to respective packages, not to th
e web server itself.\n          </p>\n        </div>\n\
n\n\n\n      </div>\n    </div>\n    <div class="valida
tor">\n    </div>\n  </body>\n</html>\n\n'
```

FIGURE 10: GET SERVER-SIDE OUTPUT

```
input a command: GET /HTTP/1.1

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transition
al//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transit
ional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
  <!--
    Modified from the Debian original for Ubuntu
    Last updated: 2014-03-19
    See: https://launchpad.net/bugs/1288690
  -->
  <head>
    <meta http-equiv="Content-Type" content="text/html;
 charset=UTF-8" />
    <title>Apache2 Ubuntu Default Page: It works</title
>
    <style type="text/css" media="screen">
  * {
    margin: 0px 0px 0px 0px;
    padding: 0px 0px 0px 0px;
    padding: 0px 0px 0px 0px;
  }

  body, html {
    padding: 3px 3px 3px 3px;

    background-color: #D8DBE2;

    font-family: Verdana, sans-serif;
    font-size: 11pt;
    text-align: center;
  }

  div.main_page {
    position: relative;
    display: table;

    width: 800px;

    margin-bottom: 3px;
    margin-left: auto;
    n-right: auto;
  }

  div.main_page {
    position: relative;
    display: table;

    width: 800px;

    margin-bottom: 3px;
    margin-left: auto;
    margin-right: auto;
    padding: 0px 0px 0px 0px;

    border-width: 2px;
    border-color: #212738;
    border-style: solid;

    background-color: #FFFFFF;

    text-al
```

Trash

FIGURE 11: GET CLIENT-SIDE OUTPUT

4) GET / folder1/test.html HTTP/1.0

```
        elif(input1 == "GET /home/seed/Networks_Project/abc.txt"):
                try:
                        receive = requests.get("http://localhost")
                        with open("/home/seed/Networks_Project/
abc.txt", "r") as f:
                                f.write(receive.content)
                                print(receive.content)
                                connectionSocket.send
(receive.content)
                except Exception as e:
                        template = "An exception of type {0}
occured. Arguments:\n{1!r}"
                        message = template.format(type(e).__name__,
e.args)
                        print(message)
                        connectionSocket.send(message.encode
('utf-8'))
```

FIGURE 12: GET FOLDER CODE

```
[02/28/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:GET /home/seed/Networks_Project/abc.txt
('From Server:', u"An exception of type IOError occured
. Arguments:\n(2, 'No such file or directory')")
```

FIGURE 13:GET SERVER-SIDE OUTPUT

```
The server is ready to receive
An exception of type IOError occured. Arguments:
(2, 'No such file or directory')
```

FIGURE 14:GET CLIENT-SIDE OUTPUT

As per the instructions given, in this example we assume there is no such file on the server. Thus, server checks that there is no such existence of the file and responds 'No such file or directory.

5) echo hello world > /home/seed/Networks_Project/output.txt

```
        elif(input1.startswith('echo')):
                print(input1[5:])
                st = input1[5:]
                print("st",st)
                m = st.index(">")
                text = st[:m]
                n = st[m:]
                print("text",text) #text
                print("n",n)
                filename = n[2:]
                print("filename",filename) #file
                with open(filename, "w") as f:
                        f.write(text)
                        f.close()
                output = "Echoed Successfully onto the text
file"
                print(output)
                connectionSocket.send(output.encode())
```

FIGURE 15: ECHO CODE

```
[02/25/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
hello world > /home/seed/Networks_Project/output.txt
('st', u'hello world > /home/seed/Networks_Project/outp
ut.txt')
('text', u'hello world ')
('n', u'> /home/seed/Networks_Project/output.txt')
('filename', u'/home/seed/Networks_Project/output.txt')
Echoed Successfully onto the text file
```

FIGURE 16: OUTPUT SERVER-SIDE

```
[02/25/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:echo hello world > /home/seed/Networks_
Project/output.txt
('From Server:', u'Echoed Successfully onto the text fi
le')
```

FIGURE 17: OUTPUT CLIENT-SIDE

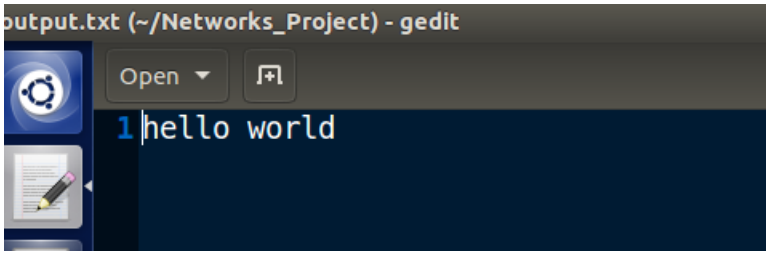

FIGURE 18: ECHO PRINT

6) ls -la filename.txt

```
        op = subprocess.Popen(str(input1), shell=True,
stderr=subprocess.PIPE,stdout=subprocess.PIPE)
        p1 = op.stdout.read()
        connectionSocket.sendall(p1)
```

FIGURE 19: LS -LA CODE

```
[02/28/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:ls -la /home/seed/Networks_Project/outp
ut.txt
('From Server:', u'-rw-rw-r-- 1 seed seed 11321 Feb 28
16:56 /home/seed/Networks_Project/output.txt\n')
```

FIGURE 20:LS -LA CLIENT SIDE OUTPUT

7) cat filename.txt

```
elif(input1.startswith('cat')):
        #"/home/seed/Networks_Project/cat.py"
        print("input", input1[4:])
        with open(input1[4:], 'r') as f:
                contents = f.read()
        print contents
        connectionSocket.send(contents.encode())
```

FIGURE 21: CAT CODE

```
('input', u'/home/seed/Networks_Project/cat.py')
import sys
with open(sys.argv[1], 'r') as f:
    contents = f.read()
print contents
```

FIGURE 22: SERVER-SIDE OUTPUT

```
[02/24/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:cat /home/seed/Networks_Project/cat.py
('From Server:', u"import sys\nwith open(sys.argv[1], '
r') as f:\n    contents = f.read()\nprint contents\n")
[02/24/20]seed@VM:~/Networks_Project$
```

FIGURE 23: CLIENT-SIDE OUTPUT

8) mkdir folder1

```
elif(input1.startswith('mkdir')):

        # Directory
        print("input",input1[6:])
        directory = input1[6:]
        #directory = "test"

        # Parent Directory path
        parent_dir = "/home/seed/Networks_Project"

        # Path
        path = os.path.join(parent_dir, directory)

        # Create the directory

        os.mkdir(path)
        print("Directory '%s' created" %directory)
        #print(output)
        connectionSocket.send(directory.encode())
```

FIGURE 24: MKDIR CODE

```
[02/24/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
('input', u'test')
Directory 'test' created
```

FIGURE 25: SERVER-SIDE OUTPUT

```
[02/24/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:mkdir test
('From Server:', u'test')
[02/24/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
```

FIGURE 26: CLIENT-SIDE OUTPUT

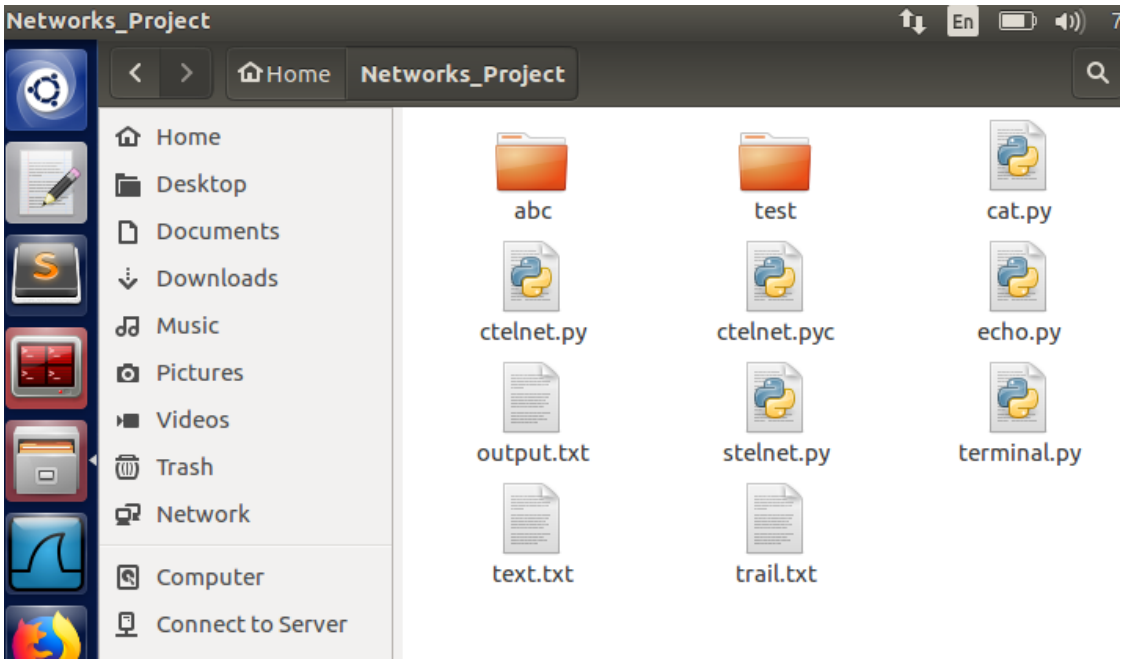

FIGURE 27: FOLDER TEST CREATED

9) rmdir folder1

```
        elif(input1.startswith('rmdir')):
                # Directory name
                print("input",input1[6:])
                directory = input1[6:]
                #directory = "test"

                # Parent Directory
                parent = "/home/seed/Networks_Project"

                # Path
                path = os.path.join(parent, directory)

                # Remove the Directory

                os.rmdir(path)
                print("Directory '%s' has been removed
successfully" %directory)
                connectionSocket.send(directory.encode())
```

FIGURE 28: RMDIR CODE

```
('input', u'test')
Directory 'test' has been removed                              s
uccessfully
```

FIGURE 29:RMDIR SERVER-SIDE OUTPUT

```
Enter a command:rmdir test
('From Server:', u'test')
[02/24/20]seed@VM:~/Networks_Project$
```

FIGURE 30: CLIENT-SIDE OUTPUT

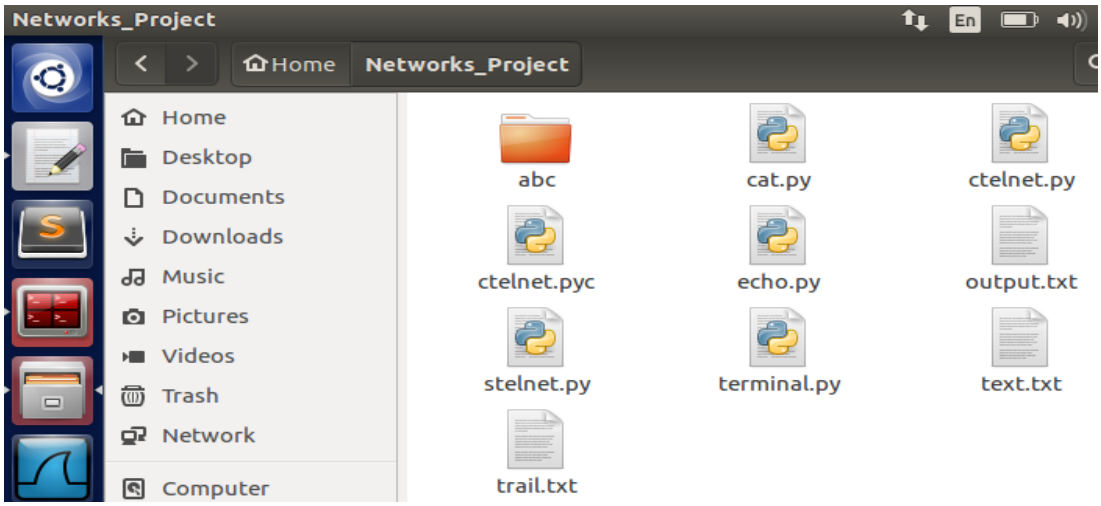

FIGURE 31: FOLDER TEST REMOVED

10) mv filename.txt folder1

```
        elif(input1.startswith('mv2')):
                print("input",input1[4:])
                st = input1[4:]
                print("st",st)
                m = st.index(" ")
                source = st[:m]
                n = st[m:]
                print("source",source)
                #print("n",n)
                destination = n[1:]
                print("destination",destination)
                #shutil.move("/home/seed/Networks_Project/text.txt", "/home/
seed/")
                shutil.move(source,destination)
                output = "File Successfully Moved"
                print(output)
                connectionSocket.send(output.encode())

        elif(input1 == "cd .."):
                output = os.chdir('..')
                output = "Directory Changed"
                connectionSocket.send(output.encode())
```

FIGURE 32: MV FILE TO FOLDER CODE

```
('input', u'/home/seed/Networks_Project/text.txt /home/
seed')
('st', u'/home/seed/Networks_Project/text.txt /home/see
d')
('source', u'/home/seed/Networks_Project/text.txt')
('destination', u'/home/seed')
File Successfully Moved
```

FIGURE 33: SERVER-SIDE OUTPUT

```
[02/25/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:mv2 /home/seed/Networks_Project/text.tx
t /home/seed
('From Server:', u'File Successfully Moved')
[02/25/20]seed@VM:~/Networks_Project$
```

FIGURE 34: CLIENT-SIDE OUTPUT

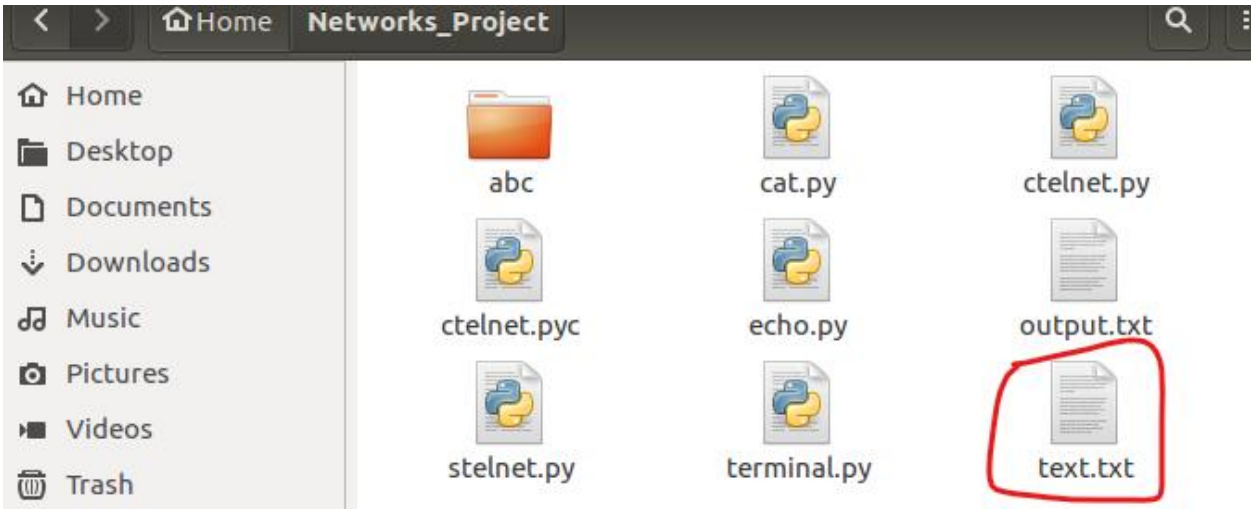

FIGURE 35: MOVING TEXT.TXT

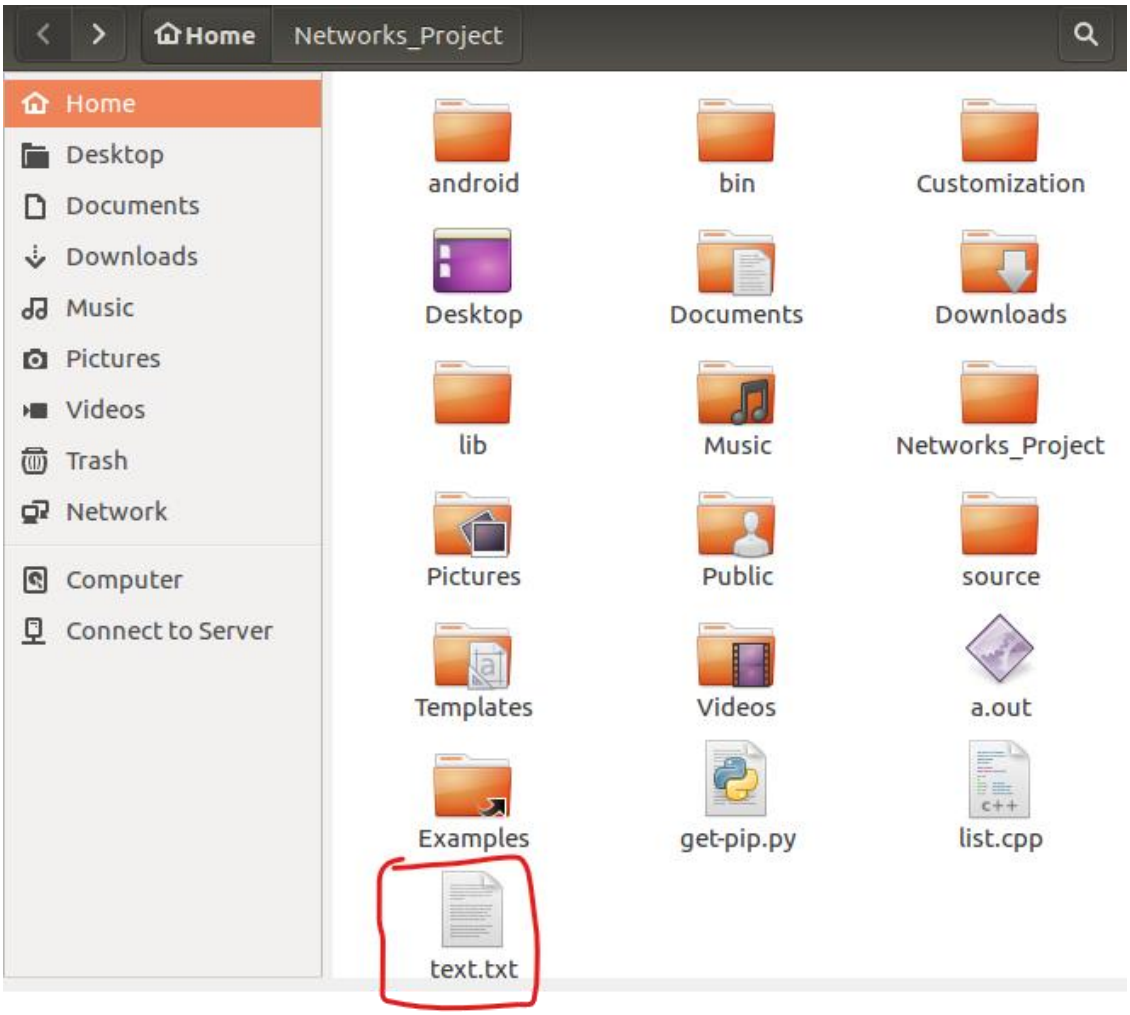

FIGURE 36: TEXT.TXT MOVED TO /HOME/SEED

11) cd folder1

```
elif(input1.startswith('cd')):
        print("input",input1[3:])
        #output = ftp.cwd(input1[3:])
        output = os.chdir(input1[3:])
        #('/home/seed/Networks_Project/abc/')
        output1 = "Directory Changed"
        connectionSocket.send(output1.encode())
```

FIGURE 37: CD CODE

```
[02/25/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
/home/seed/Networks_Project
('input', u'abc')
/home/seed/Networks_Project/abc
```

FIGURE 38: SERVER-SIDE OUTPUT

```
Enter a command:pwd
('From Server:', u'/home/seed/Networks_Project')
[02/25/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:cd abc
('From Server:', u'Directory Changed')
[02/25/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:pwd
('From Server:', u'/home/seed/Networks_Project/abc')
[02/25/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
```

FIGURE 39: CLIENT-SIDE OUTPUT SHOWING THE CHANGE IN DIRECTORY

12) pwd

```
elif(input1== "pwd"):
        currentDirectory = os.getcwd()
        #output = ftp.pwd()
        output = ftp.mkd('/Networks_Project/TempDir')
        print(currentDirectory)
        connectionSocket.send(currentDirectory.encode())
```

FIGURE 40: PWD CODE

```
Enter a command:pwd
('From Server:', u'/home/seed/Networks_Project/abc')
[02/25/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
```

FIGURE 41:PWD OUTPUT

13) ls -la

```
elif (input1 == "ls -la"):
        output = ftp.retrlines('LIST')
        print(output)
        connectionSocket.send(output.encode())
```

FIGURE 42:LS -LA CODE

```
drwxrwxr-x    2 1000     1000         4096 Jan 14  2018
 Customization
drwxr-xr-x    2 1000     1000         4096 Feb 19 19:51
 Desktop
drwxr-xr-x    2 1000     1000         4096 Jul 25  2017
 Documents
drwxr-xr-x    3 1000     1000         4096 Feb 19 14:45
 Downloads
drwxr-xr-x    2 1000     1000         4096 Jul 25  2017
 Music
drwxrwxrwx    3 1000     1000         4096 Feb 24 19:53
 Networks_Project
drwxr-xr-x    3 1000     1000         4096 Jan 14  2018
 Pictures
drwxr-xr-x    2 1000     1000         4096 Jul 25  2017
 Public
drwxr-xr-x    2 1000     1000         4096 Jul 25  2017
 Templates
drwxr-xr-x    2 1000     1000         4096 Jul 25  2017
 Videos
-rwxrwxr-x    1 1000     1000        15664 Feb 19 14:43
drwxrwxr-x    4 1000     1000         4096 Apr 30  2018
 android
drwxrwxr-x    2 1000     1000         4096 Jan 14  2018
 bin
-rw-r--r--    1 1000     1000         8980 Jul 25  2017
 examples.desktop
-rw-rw-r--    1 1000     1000      1661676 Jan 02  2019
 get-pip.py
drwxrwxr-x    3 1000     1000         4096 May 08  2018
 lib
-rw-rw-r--    1 1000     1000          703 Feb 19 14:43
 list.cpp
drwxrwxr-x    4 1000     1000         4096 May 08  2018
 source
226 Directory send OK.
```

FIGURE 43: LS -LA OUTPUT

```
[02/24/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:ls -la
('From Server:', u'226 Directory send OK.226 Directory
send OK.')
[02/24/20]seed@VM:~/Networks_Project$
```

FIGURE 44: LS -LA CLIENT-SIDE OUTPUT

14) At this point, open a terminal at the server and make sure that folder1 has been created, and the file is in it. Also cat the file to check its contents. In your screenshot, highlight the server name.(for us folder1 is abc folder)

```
elif(input1.startswith('cat')):
        #"/home/seed/Networks_Project/cat.py"
        print("input", input1[4:])
        with open(input1[4:], 'r') as f:
                contents = f.read()
        print contents
        connectionSocket.send(contents.encode())
```

FIGURE 45: CODE FOR CAT COMMAND

```
[02/28/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
('input', u'/home/seed/Networks_Project/abc/')
Hello! This is a file!
```

FIGURE 46:SERVER-SIDE OUTPUT – DISPLAY FILE CONTENTS USING CAT COMMAND

```
[02/27/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:cd /home/seed/Networks_Project/abc/
('From Server:', u'Directory Changed')
[02/27/20]seed@VM:~/Networks_Project$ pwd
/home/seed/Networks_Project
[02/27/20]seed@VM:~/Networks_Project$ cat /home/seed/Ne
tworks_Project/abc/filename.txt
Hello! This is a file!
[02/27/20]seed@VM:~/Networks_Project$
```

FIGURE 47: CONTENT OF FILENAME .TXT DISPLAYED USING CAT

127.0.0.1 is IP address of our server.

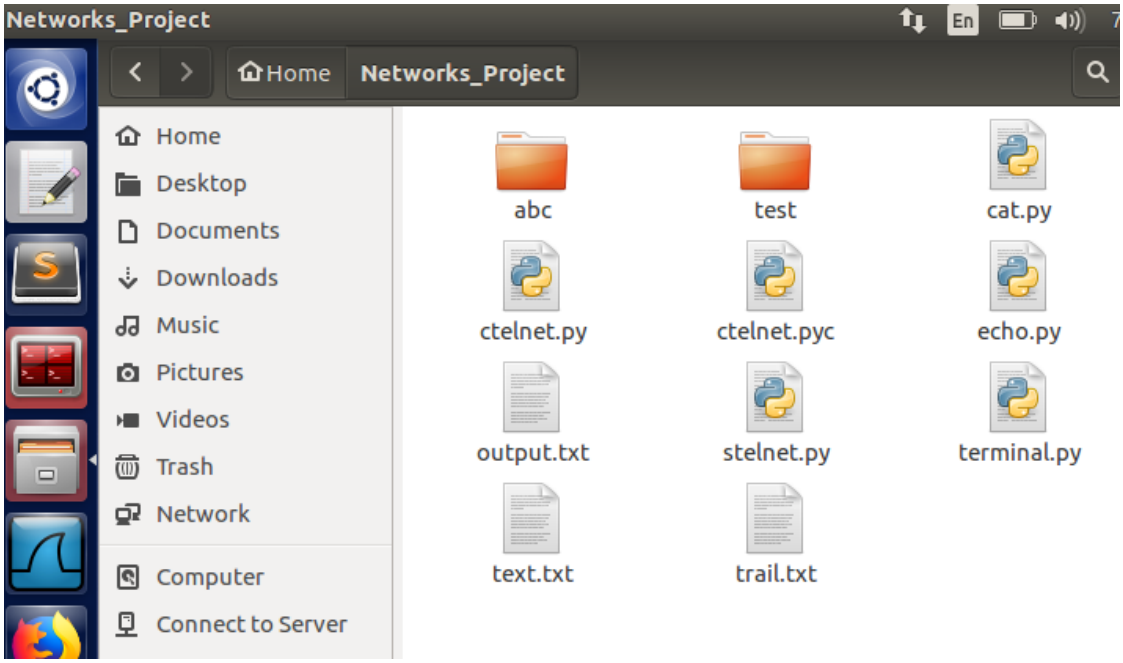

FIGURE 48: FOLDER ABC CREATED

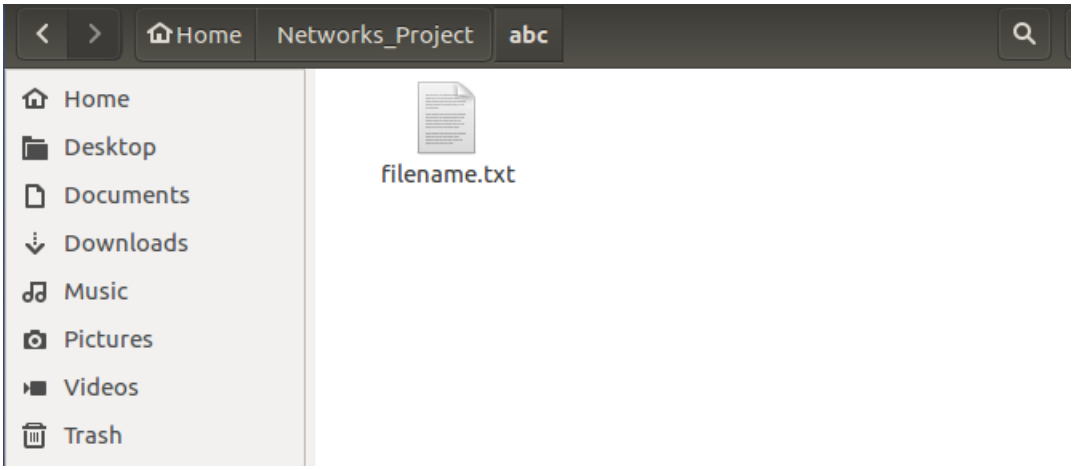

FIGURE 49:FILENAME .TXT EXISTS INSIDE FOLDER ABC

15) mv filename.txt file1

```
        elif(input1.startswith('mv1')):
                print("input",input1[4:])
                st = input1[4:]
                print("st",st)
                m = st.index(" ")
                source = st[:m]
                n = st[m:]
                print("source",source)
                #print("n",n)
                destination = n[1:]
                print("destination",destination)

                #os.rename("/home/seed/Networks_Project/trial.txt", "/home/
seed/text.txt")
                os.rename(source,destination)
                output = "File Successfully Moved"
                print(output)
                connectionSocket.send(output.encode())
```

FIGURE 50: MOVE CONTENTS FROM FILE1 TO FILE 2

```
[02/25/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
('input', u'/home/seed/Networks_Project/trial.txt /home
/seed/Networks_Project/text.txt')
('st', u'/home/seed/Networks_Project/trial.txt /home/se
ed/Networks_Project/text.txt')
('source', u'/home/seed/Networks_Project/trial.txt')
('destination', u'/home/seed/Networks_Project/text.txt'
)
File Successfully Moved
```

FIGURE 51: SERVER-SIDE OUTPUT

```
[02/25/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:mv1 /home/seed/Networks_Project/trial.t
xt /home/seed/Networks_Project/text.txt
('From Server:', u'File Successfully Moved')
[02/25/20]seed@VM:~/Networks_Project$
```

FIGURE 52: CLIENT-SIDE OUTPUT

The file named trial.txt has been moved to text.txt successfully

16) cat file1

```
elif(input1.startswith('cat')):
        #"/home/seed/Networks_Project/cat.py"
        print("input", input1[4:])
        with open(input1[4:], 'r') as f:
                contents = f.read()
        print contents
        connectionSocket.send(contents.encode())
```

FIGURE 53: CAT CODE

```
[02/26/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
('input', u'/home/seed/Networks_Project/file1')
This is file 1
```

FIGURE 54: SEVER SIDE OUTPUT

```
[02/26/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:cat /home/seed/Networks_Project/file1
('From Server:', u'This is file 1\n')
[02/26/20]seed@VM:~/Networks_Project$
```

FIGURE 55: CAT DISPLAYS CONTENTS OF FILE

17) rm file1

```
elif(input1.startswith('rm')):
        #"/home/seed/Networks_Project/text.txt"
        print("input", input1[3:])
        os.remove(input1[3:])
        output = "File Successfully Removed"
        print(output)
        connectionSocket.send(output.encode())
```

FIGURE 56: RM CODE

```
[02/24/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
('input', u'/home/seed/Networks_Project/trial.txt')
File Successfully Removed
```

FIGURE 57: RM SERVER-SIDE OUTPUT

```
[02/24/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:rm /home/seed/Networks_Project/trial.tx
t
('From Server:', u'File Successfully Removed')
```

FIGURE 58: RM CLIENT-SIDE OUTPUT

18) ls

```
input1 = connectionSocket.recv(1024).decode()
if (input1 == "ls"):
        output = ftp.nlst()
        print(output)
        output1 = "Contents Listed"
        connectionSocket.send(output1.encode())
```

FIGURE 59:LS CODE

```
[02/24/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
['Customization', 'Desktop', 'Documents', 'Downloads',
'Music', 'Networks_Project', 'Pictures', 'Public', 'Tem
plates', 'Videos', 'a.out', 'android', 'bin', 'examples
.desktop', 'get-pip.py', 'lib', 'list.cpp', 'source']
```

FIGURE 60: LS OUTPUT

19) At this point, open a terminal at the server and make sure that folder1 has no file anymore.

```
elif(input1.startswith('rm')):
        #"/home/seed/Networks_Project/text.txt"
        print("input", input1[3:])
        os.remove(input1[3:])
        output = "File Successfully Removed"
        print(output)
        connectionSocket.send(output.encode())
```

FIGURE 61: REMOVE THE TRIAL.TXT FILE

```
[02/24/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
('input', u'/home/seed/Networks_Project/trial.txt')
File Successfully Removed
```

FIGURE 62: SERVER SIDE OUTPUT

```
[02/24/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:rm /home/seed/Networks_Project/trial.tx
t
('From Server:', u'File Successfully Removed')
```

FIGURE 63: REMOVED FILE

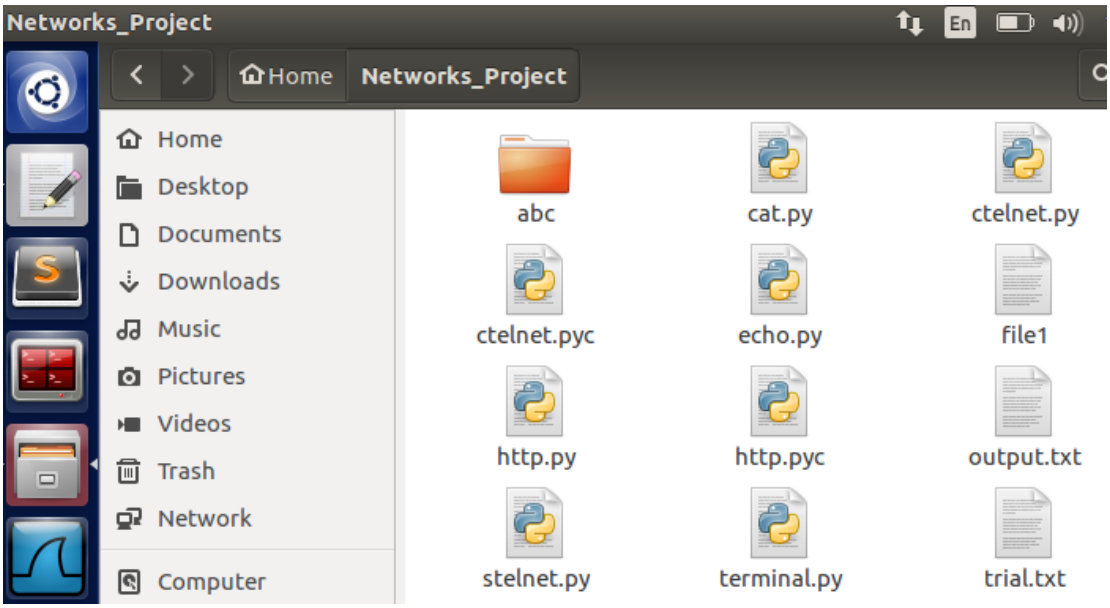

FIGURE 64: BEFORE REMOVING TRIAL.TXT

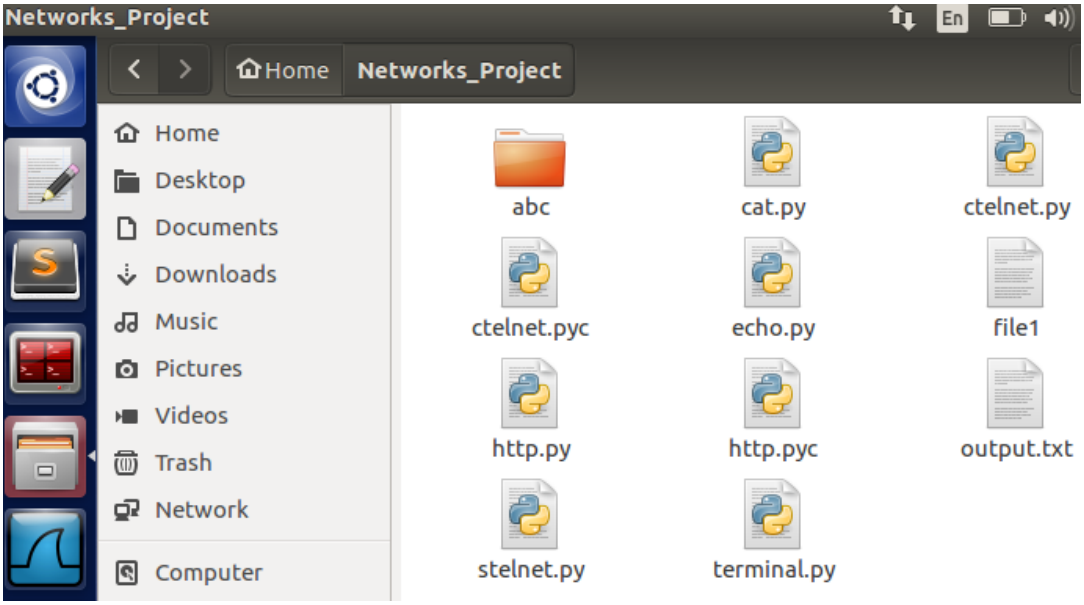

FIGURE 65: AFTER REMOVING TRIAL.TXT

The folder Networks_Project doesn't have the file trial.txt anymore, has been successfully removed.

20) cd..

```
elif(input1.startswith('cd')):
        print("input",input1[3:])
        #output = ftp.cwd(input1[3:])
        output = os.chdir(input1[3:])
        #('/home/seed/Networks_Project/abc/')
        output1 = "Directory Changed"
        connectionSocket.send(output1.encode())
```

FIGURE 66: CD.. CODE

```
/home/seed/Networks_Project/abc
('input', u'..')
/home/seed/Networks_Project
```

FIGURE 67: CD.. SERVER-SIDE OUTPUT

```
Enter a command:cd ..
('From Server:', u'Directory Changed')
[02/25/20]seed@VM:~/Networks_Project$ python ctelnet.py
 127.0.0.1 10100
Enter a command:pwd
('From Server:', u'/home/seed/Networks_Project')
[02/25/20]seed@VM:~/Networks_Project$
```

FIGURE 68: CD.. CLIENT-SIDE OUTPUT

21) ls

```
input1 = connectionSocket.recv(1024).decode()
if (input1 == "ls"):
        output = ftp.nlst()
        print(output)
        output1 = "Contents Listed"
        connectionSocket.send(output1.encode())
```

FIGURE 69: LS CODE

```
[02/24/20]seed@VM:~/Networks_Project$ python stelnet.py

The server is ready to receive
['Customization', 'Desktop', 'Documents', 'Downloads',
'Music', 'Networks_Project', 'Pictures', 'Public', 'Tem
plates', 'Videos', 'a.out', 'android', 'bin', 'examples
.desktop', 'get-pip.py', 'lib', 'list.cpp', 'source']
```

FIGURE 70: LS OUTPUT