

LABORATORY PROGRAM – 3

Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of the requested file if present.

(P3)
Using TCP/IP sockets, write a client-server program to make client sending the file name and the server to send back the contents of requested file if present.

```
① Client TCP.py:  
from socket import *  
ServerName = '127.0.0.1'  
ServerPort = 12000  
clientSocket = socket(AF_INET, SOCK_STREAM)  
clientSocket.connect((ServerName, ServerPort))  
Sentence = input('Enter file name:')  
clientSocket.send(Sentence.encode())  
filecontents = clientSocket.recv(1024).decode()  
print('From Server:')  
print(filecontents)  
clientSocket.close()
```

Output:

python Server TCP.py	python Client
① The Server is ready to receive	① Enter file name:
② Connection established with ('127.0.0.1', 60477)	Sample.txt
③ Sent contents of Sample.txt	③ From Server:
④ Server is ready to receive	This is a Sample Text file
	④ # close

Code: Client.py

```
from socket import *
serverName = "127.0.0.1" # Server address (localhost)
serverPort = 12000 # Port number where the server listens

# Create TCP socket
clientSocket = socket(AF_INET, SOCK_STREAM)
clientSocket.connect((serverName, serverPort)) # Connect to server

# Ask user for file name to request
sentence = input("Enter file name: ")

# Send file name to server
clientSocket.send(sentence.encode())

# Receive file contents from server
filecontents = clientSocket.recv(1024).decode()
print('From Server:', filecontents)

# Close the connection
clientSocket.close()
```

② ServerTCP.py

```
from Socket import *
```

```
ServerName = '127.0.0.1'
```

```
ServerPort = 12000
```

```
ServerSocket = Socket(AF_INET, SOCK_STREAM)
```

```
ServerSocket.bind((ServerName, ServerPort))
```

```
ServerSocket.listen(1)
```

```
while 1:
```

```
    print('The Server is ready to receive')
```

```
    ConnectionSocket, address = ServerSocket.accept()
```

```
    print(f'Connection established with {address}')
```

```
    Sentence = ConnectionSocket.recv(1024).decode()
```

```
    file = open(Sentence, 'r')
```

```
    d = file.read(1024)
```

```
    ConnectionSocket.send(d.encode())
```

```
    print(f'Sent contents of {Sentence}')
```

```
    file.close()
```

```
    ConnectionSocket.close()
```

#1024 - no. of bytes that the Server will try to send at once from Socket.

Code: Server.py

```
from socket import *
serverName = "127.0.0.1" # Server address (localhost)
serverPort = 12000 # Port number to listen on

# Create TCP socket
serverSocket = socket(AF_INET, SOCK_STREAM)
serverSocket.bind((serverName, serverPort)) # Bind socket to the address and port
serverSocket.listen(1) # Listen for 1 connection
print("The server is ready to receive")

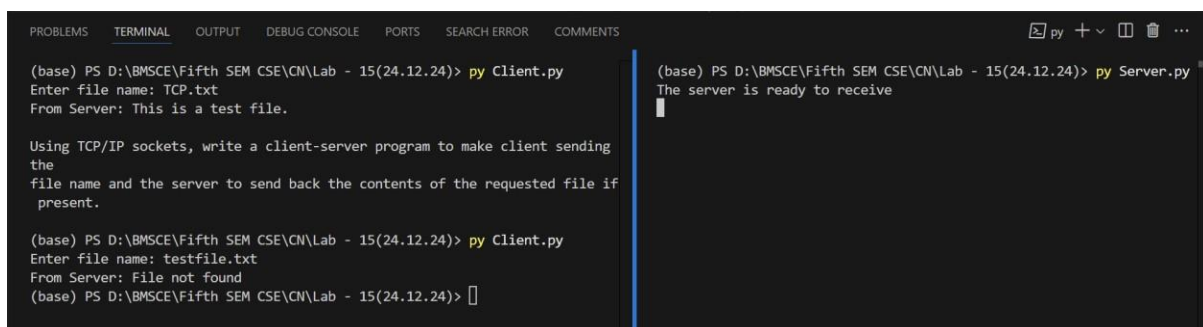
while True:
    # Accept a connection
    connectionSocket, addr = serverSocket.accept()

    # Receive the file name from the client
    sentence = connectionSocket.recv(1024).decode()

    # Try opening the file
    try:
        file = open(sentence, "r") # Open file in read mode
        fileContents = file.read(1024) # Read file content (up to 1024 bytes)
        connectionSocket.send(fileContents.encode()) # Send file contents to client
        file.close()
    except FileNotFoundError:
        # Send error message if file not found
        connectionSocket.send("File not found".encode())

    # Close the connection
    connectionSocket.close()
```

Output



```
PROBLEMS  TERMINAL  OUTPUT  DEBUG CONSOLE  PORTS  SEARCH ERROR  COMMENTS
(base) PS D:\BMSCE\Fifth SEM CSE\CN\Lab - 15(24.12.24)> py Client.py
Enter file name: TCP.txt
From Server: This is a test file.

Using TCP/IP sockets, write a client-server program to make client sending
the
file name and the server to send back the contents of the requested file if
present.

(base) PS D:\BMSCE\Fifth SEM CSE\CN\Lab - 15(24.12.24)> py Client.py
Enter file name: testfile.txt
From Server: File not found
(base) PS D:\BMSCE\Fifth SEM CSE\CN\Lab - 15(24.12.24)>

(base) PS D:\BMSCE\Fifth SEM CSE\CN\Lab - 15(24.12.24)> py Server.py
The server is ready to receive
```