# Weekly Assessment - 9

Thanvi Katakam
2023006366

Aim:

To implement a file transfer system using socket programming in Python, where a client requests a file from the server, and the server sends the file content back to the client.

Description:

Socket programming enables communication between processes over a network using the TCP/IP protocol stack. In this project, we create a server that listens for file requests and a client that requests files from the server. If the file exists, the server sends its content; otherwise, it informs the client that the file is not found.

**Procedure:**

**Step 1: Create the Server Code**

1. Open **Notepad**.
2. Write the server code:
3. Click File and then click Save As
4. Save the file as server.py, selecting All Files as the file type.

Step 2: Create the Client Code

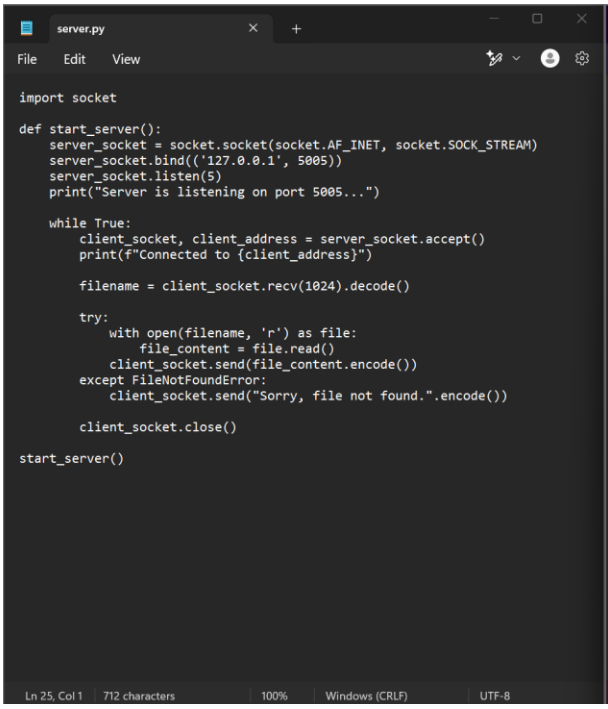
1. Open Notepad.
2. Write the client code:

**3.Click '**File' and then we click 'Save As'

4.Save the file as **client.py**, selecting **All Files** as the file type.

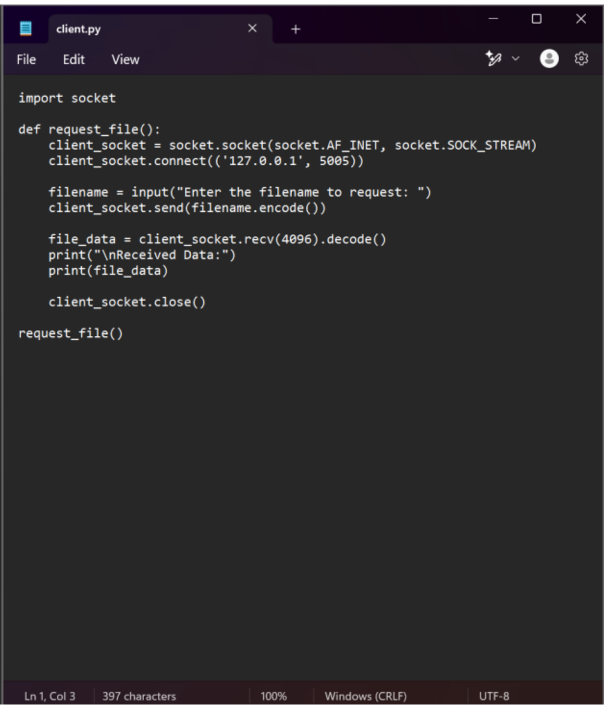
## Code:

### Server Code (server.py):



```
import socket

def start_server():
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    server_socket.bind(('127.0.0.1', 5005))
    server_socket.listen(5)
    print("Server is listening on port 5005...")

    while True:
        client_socket, client_address = server_socket.accept()
        print(f"Connected to {client_address}")

        filename = client_socket.recv(1024).decode()

        try:
            with open(filename, 'r') as file:
                file_content = file.read()
            client_socket.send(file_content.encode())
        except FileNotFoundError:
            client_socket.send("Sorry, file not found.".encode())

        client_socket.close()

start_server()
```

### Client Code (client.py):

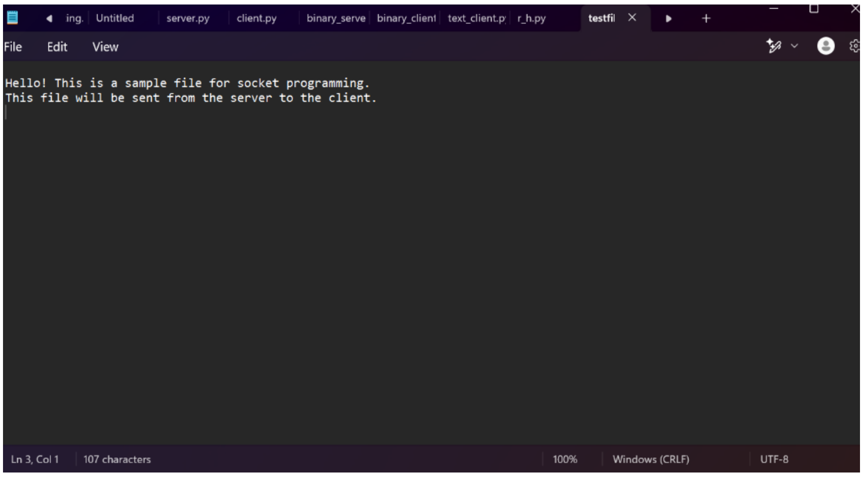


```
import socket

def request_file():
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect(('127.0.0.1', 5005))

    filename = input("Enter the filename to request: ")
    client_socket.send(filename.encode())

    file_data = client_socket.recv(4096).decode()
    print("\nReceived Data:")
    print(file_data)

    client_socket.close()

request_file()
```

Step 3: Create a Sample File for Testing

1. Open Notepad.
2. Type the following content:
3. Click ‘File’ then click ‘Save As’
4. Save the file as testfile.txt, selecting All Files as the file type.

### Test File(Test file.txt):



```
Hello! This is a sample file for socket programming.
This file will be sent from the server to the client.
```

Step 4: Run the Server

1. Open Command Prompt (cmd).
2. Navigate to the folder where you saved the files:
3. Run the server->’server.py’

Step 5: Run the Client to Request a File

1. Open a new Command Prompt window.
2. Navigate to the same folder:
3. ->Run the client->’client.py’

→ When prompted, enter the filename to request:

- ‘testfile.txt’

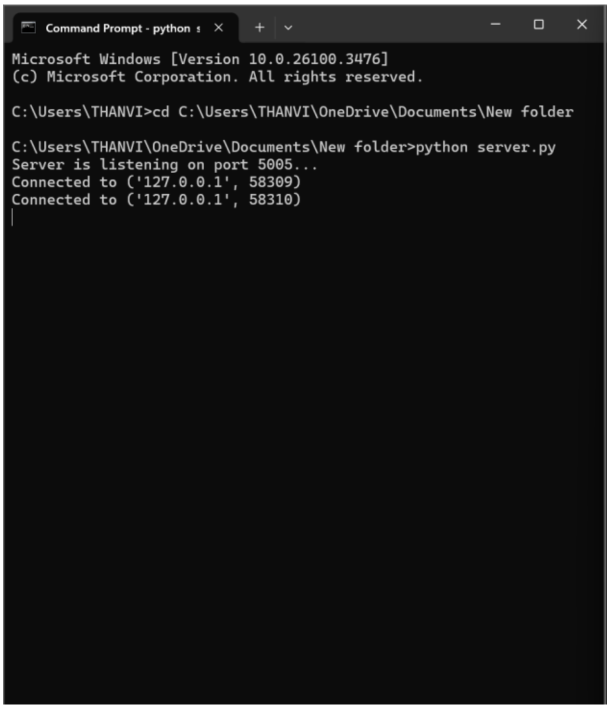
  (it will print “Hello! This is a sample file for socket programming.

  This file will be sent from the server to the client.”)

- If we type anything else other than ‘testfile.txt’

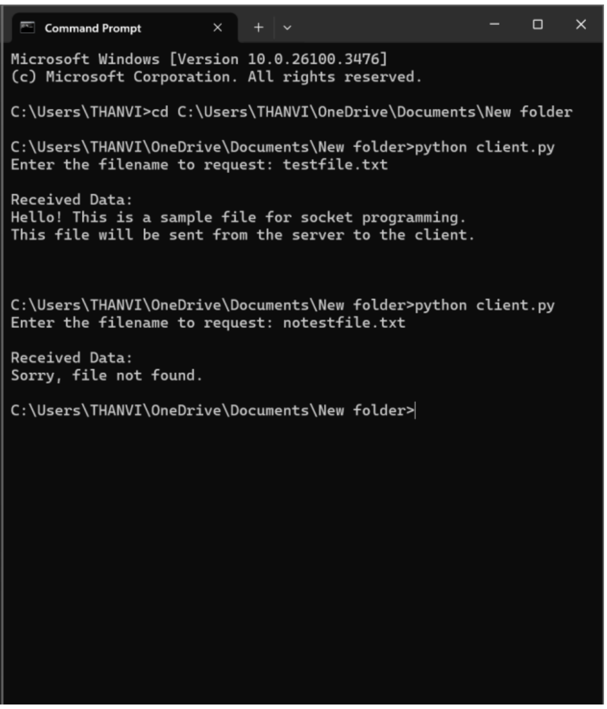
  ( Sorry, file not found.)

Server Output:



```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\THANVI>cd C:\Users\THANVI\OneDrive\Documents\New folder

C:\Users\THANVI\OneDrive\Documents\New folder>python server.py
Server is listening on port 5005...
Connected to ('127.0.0.1', 58309)
Connected to ('127.0.0.1', 58310)
```

Client Output:

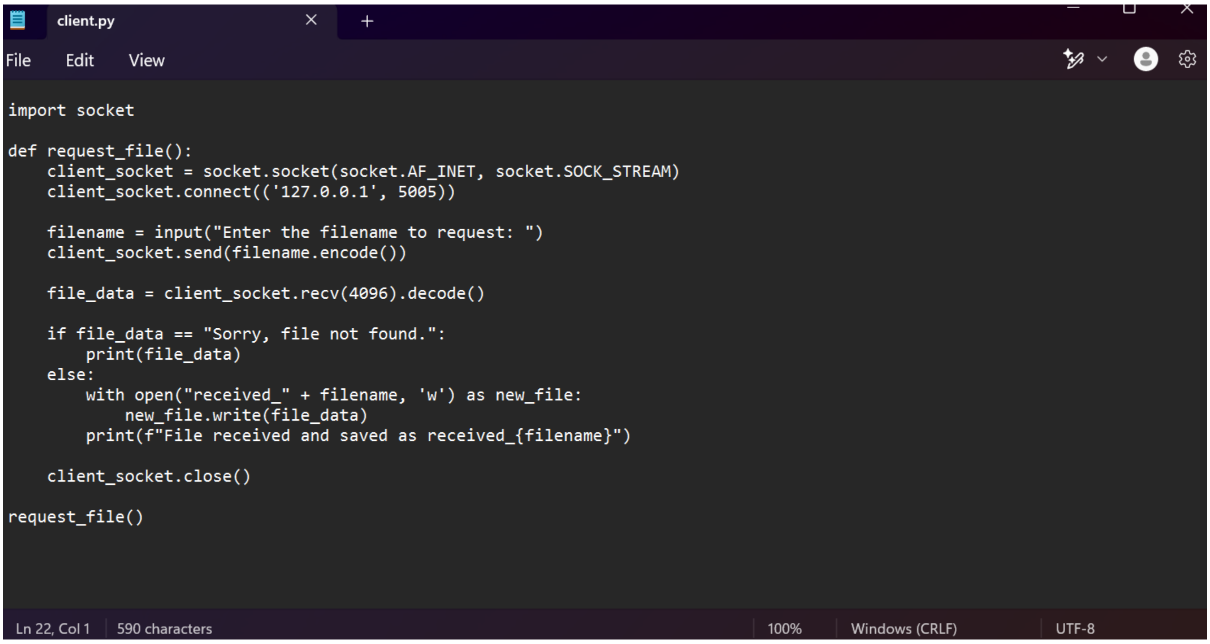


```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\THANVI>cd C:\Users\THANVI\OneDrive\Documents\New folder

C:\Users\THANVI\OneDrive\Documents\New folder>python client.py
Enter the filename to request: testfile.txt

Received Data:
Hello! This is a sample file for socket programming.
This file will be sent from the server to the client.


C:\Users\THANVI\OneDrive\Documents\New folder>python client.py
Enter the filename to request: notestfile.txt

Received Data:
Sorry, file not found.

C:\Users\THANVI\OneDrive\Documents\New folder>
```

## Step 6: Verify File Saving

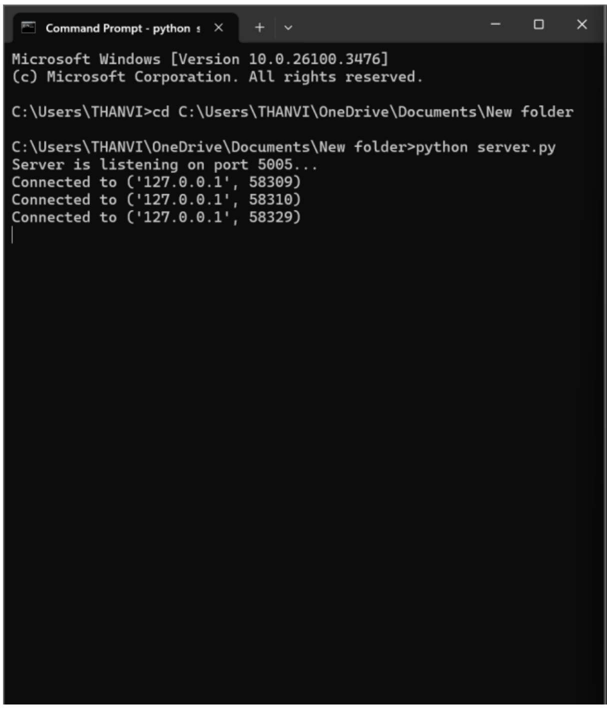
1. Modify client.py so that it saves the received file instead of just printing it.
2. Run python client.py and enter testfile.txt.
3. Check your folder for a new file named received_testfile.txt.

## Modified client code(client.py):



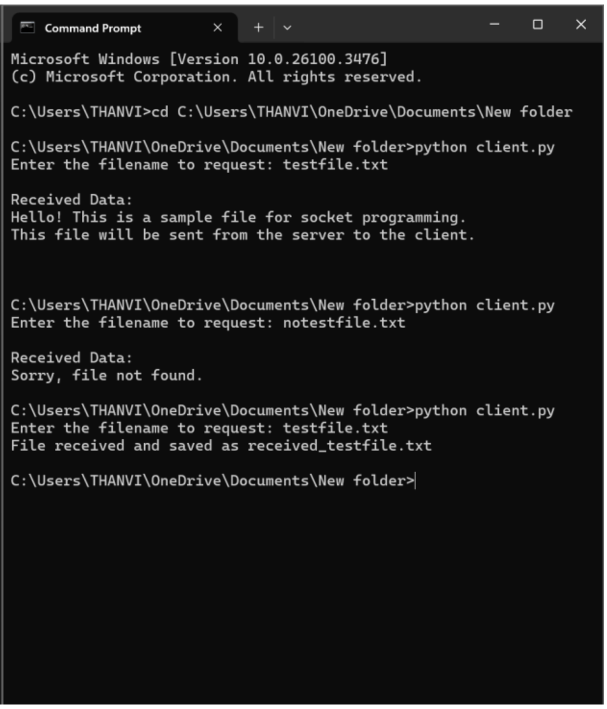
```
import socket

def request_file():
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    client_socket.connect(('127.0.0.1', 5005))

    filename = input("Enter the filename to request: ")
    client_socket.send(filename.encode())

    file_data = client_socket.recv(4096).decode()

    if file_data == "Sorry, file not found.":
        print(file_data)
    else:
        with open("received_" + filename, 'w') as new_file:
            new_file.write(file_data)
        print(f"File received and saved as received_{filename}")

    client_socket.close()

request_file()
```

Server output:



```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\THANVI>cd C:\Users\THANVI\OneDrive\Documents\New folder

C:\Users\THANVI\OneDrive\Documents\New folder>python server.py
Server is listening on port 5005...
Connected to ('127.0.0.1', 58309)
Connected to ('127.0.0.1', 58310)
Connected to ('127.0.0.1', 58329)
```

Client output:



```
Microsoft Windows [Version 10.0.26100.3476]
(c) Microsoft Corporation. All rights reserved.

C:\Users\THANVI>cd C:\Users\THANVI\OneDrive\Documents\New folder

C:\Users\THANVI\OneDrive\Documents\New folder>python client.py
Enter the filename to request: testfile.txt

Received Data:
Hello! This is a sample file for socket programming.
This file will be sent from the server to the client.


C:\Users\THANVI\OneDrive\Documents\New folder>python client.py
Enter the filename to request: notestfile.txt

Received Data:
Sorry, file not found.

C:\Users\THANVI\OneDrive\Documents\New folder>python client.py
Enter the filename to request: testfile.txt
File received and saved as received_testfile.txt

C:\Users\THANVI\OneDrive\Documents\New folder>
```

## Conclusion:

This project successfully demonstrates file transfer using Python socket programming. The server listens for file requests and responds with the file content or an error message. The client receives and saves the file, enabling simple yet effective data exchange over a network.