

LAB ASSIGNMENT V
COMPUTER NETWORKS LAB (BCSE308P)
CLASS ID - VL2024250102387

MAXIMUM MARKS: 10

DUE DATE: 15 NOVEMBER 2024

NAME: R HEMESH

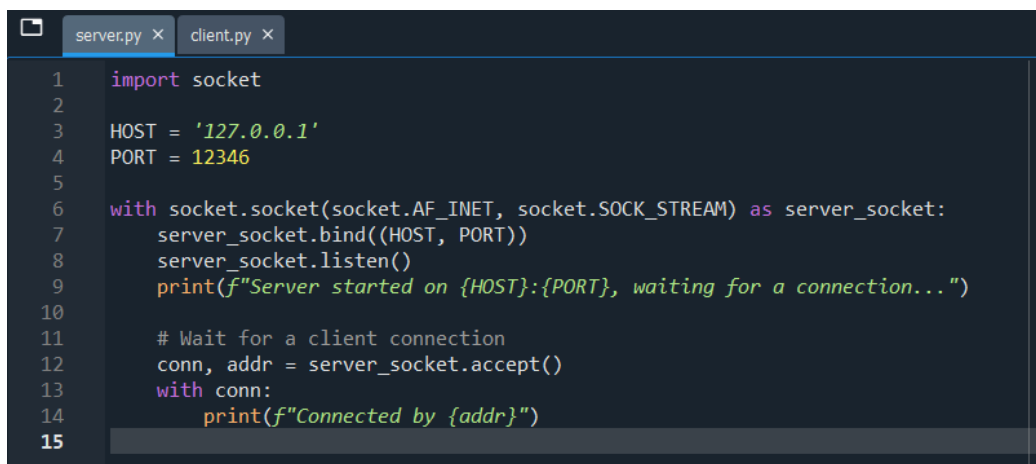
REG NO:22BCT0328

Using the Spyder IDE, perform the following tasks utilizing Socket Programming library in Python. Include the screenshot of each step.

1. The server is first started on a known port.

server.py :

```
import socket
HOST = '127.0.0.1'
PORT = 12345
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
    server_socket.bind((HOST, PORT))
    server_socket.listen()
    print(f"Server started on {HOST}:{PORT}, waiting for a connection...")
    # Wait for a client connection
    conn, addr = server_socket.accept()
    with conn:
        print(f"Connected by {addr}")
```



```
1 import socket
2
3 HOST = '127.0.0.1'
4 PORT = 12346
5
6 with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
7     server_socket.bind((HOST, PORT))
8     server_socket.listen()
9     print(f"Server started on {HOST}:{PORT}, waiting for a connection...")
10
11     # Wait for a client connection
12     conn, addr = server_socket.accept()
13     with conn:
14         print(f"Connected by {addr}")
15
```

OUTPUT in the server.py terminal:

```
In [3]: %runfile 'C:/Users/hemes/Desktop/socket prog/server.py' --wdir
Server started on 127.0.0.1:12346, waiting for a connection...
```

2. The client program is started (server IP & port are provided on the commandline).

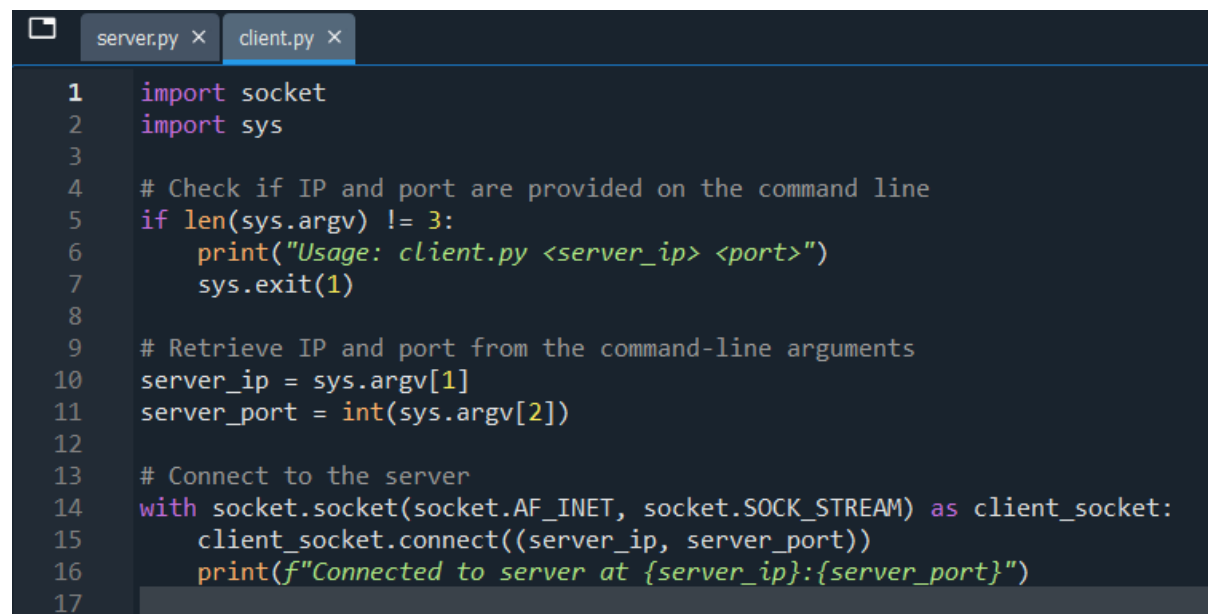
client.py :

```
import socket
import sys

# Check if IP and port are provided on the command line
if len(sys.argv) != 3:
    print("Usage: client.py <server_ip> <port>")
    sys.exit(1)

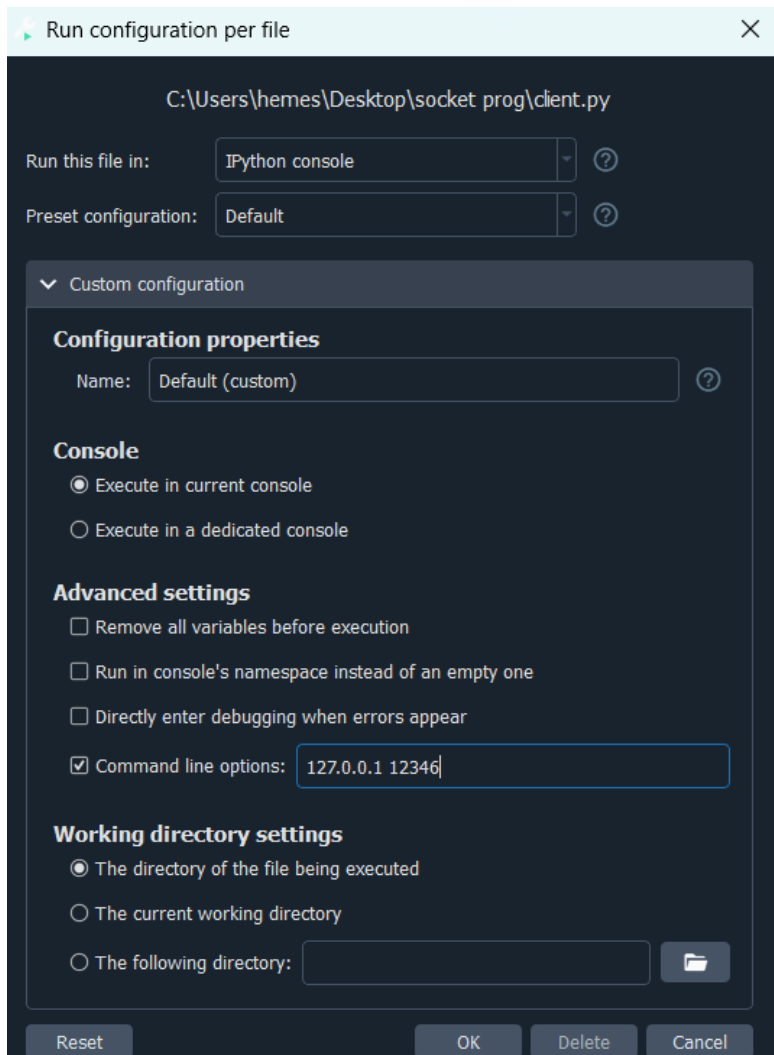
# Retrieve IP and port from the command-line arguments
server_ip = sys.argv[1]
server_port = int(sys.argv[2])

# Connect to the server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
    client_socket.connect((server_ip, server_port))
    print(f"Connected to server at {server_ip}:{server_port}")
```



The screenshot shows a code editor with two tabs: 'server.py' and 'client.py'. The 'client.py' tab is active, displaying the same Python code as the previous block. Line numbers 1 through 17 are visible on the left side of the editor.

```
1 import socket
2 import sys
3
4 # Check if IP and port are provided on the command line
5 if len(sys.argv) != 3:
6     print("Usage: client.py <server_ip> <port>")
7     sys.exit(1)
8
9 # Retrieve IP and port from the command-line arguments
10 server_ip = sys.argv[1]
11 server_port = int(sys.argv[2])
12
13 # Connect to the server
14 with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
15     client_socket.connect((server_ip, server_port))
16     print(f"Connected to server at {server_ip}:{server_port}")
17
```



server IP & port are provided on the commandline as specified.

In the configuration file of client.py , we provide server IP and port in the command line
“ 127.0.0.1 12346”

OUTPUT in the client.py terminal :

```
Python 3.11.10 | packaged by conda-forge | (main, Oct 16 2024, 01:17:14) [MSC v.1941 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more information.

IPython 8.29.0 -- An enhanced Interactive Python. Type '?' for help.

In [1]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
```

OUTPUT in the server.py terminal:

```
In [3]: %runfile 'C:/Users/hemes/Desktop/socket prog/server.py' --wdir
Server started on 127.0.0.1:12346, waiting for a connection...
Connected by ('127.0.0.1', 51041)
```

3. The client connects to the server, and then asks the user for input. The user types the message on the terminal "My name is ". The user's input is sent to the server via the connected socket.

client.py:

```
# Step 3: Ask for the user's name
name_message = input("Enter your name message: ")
client_socket.sendall(name_message.encode())
response = client_socket.recv(1024).decode()
print("Server:", response)
```

```
17
18     # Step 3: Ask for the user's name
19     name_message = input("Enter your name message: ")
20     client_socket.sendall(name_message.encode())
21     response = client_socket.recv(1024).decode()
22     print("Server:", response)
23
```

OUTPUT in the client.py terminal :

```
In [12]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
```

4. The server reads the user's input from the client socket and replies with "Hello ".
- server.py:

```
# Step 4: Receive and respond to the name message
name_message = conn.recv(1024).decode()
if name_message.startswith("My name is"):
    name = name_message.split("My name is ")[1]
    conn.sendall(f"Hello {name}".encode())
```

```
15
16     # Step 4: Receive and respond to the name message
17     name_message = conn.recv(1024).decode()
18     if name_message.startswith("My name is"):
19         name = name_message.split("My name is ")[1]
20         conn.sendall(f"Hello {name}".encode())
21
```

OUTPUT in the client.py terminal :

```
Console 2/A x Console 3/A x
In [12]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
```

**5. Then, the client asks the user for the registration number to send to the server.
client.py:**

```
# Step 5: Send registration number to the server
reg_number = input("Enter your registration number: ")
client_socket.sendall(reg_number.encode())
response = client_socket.recv(1024).decode()
print("Server:", response)
```

```
23
24     # Step 5: Send registration number to the server
25     reg_number = input("Enter your registration number: ")
26     client_socket.sendall(reg_number.encode())
27     response = client_socket.recv(1024).decode()
28     print("Server:", response)
29
```

OUTPUT in the client.py terminal :

```
Console 2/A x Console 6/A x
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
```

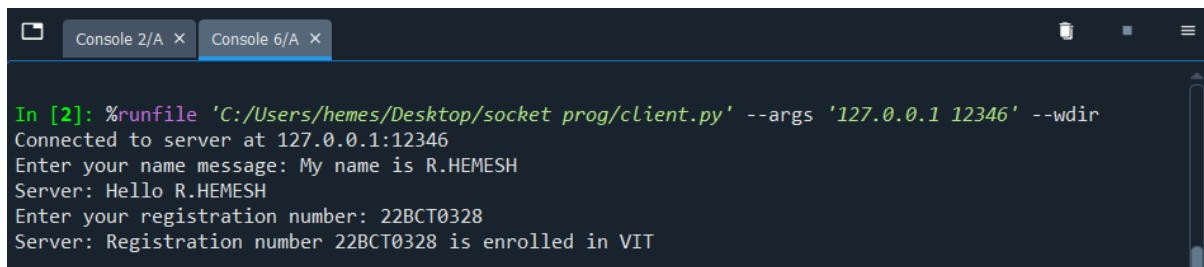
6. The server must reply with "Registration number is enrolled in VIT".

server.py:

```
# Step 6: Receive and respond to the registration number
reg_number = conn.recv(1024).decode()
conn.sendall(f"Registration number {reg_number} is enrolled in VIT".encode())
```

```
21
22     # Step 6: Receive and respond to the registration number
23     reg_number = conn.recv(1024).decode()
24     conn.sendall(f"Registration number {reg_number} is enrolled in VIT".encode())
25
```

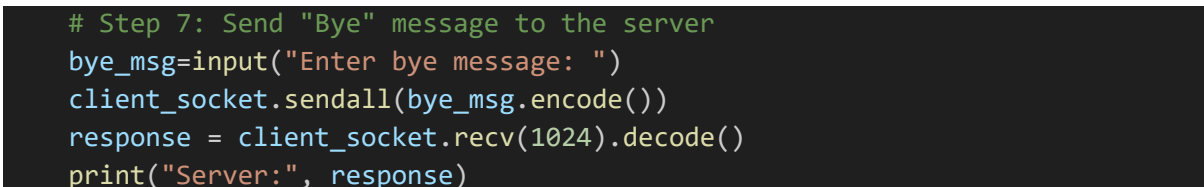
OUTPUT in the client.py terminal :



```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
```

7. Finally, the user types “Bye” and sends to the server.

client.py:

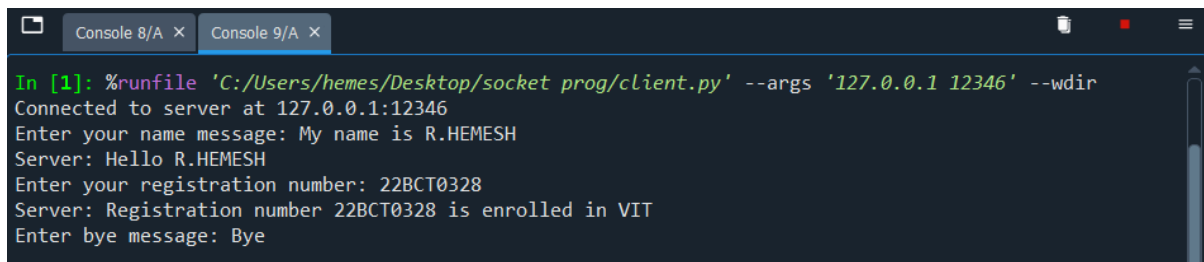


```
# Step 7: Send "Bye" message to the server
bye_msg=input("Enter bye message: ")
client_socket.sendall(bye_msg.encode())
response = client_socket.recv(1024).decode()
print("Server:", response)
```



```
29
30     # Step 7: Send "Bye" message to the server
31     bye_msg=input("Enter bye message: ")
32     client_socket.sendall(bye_msg.encode())
33     response = client_socket.recv(1024).decode()
34     print("Server:", response)
35
```

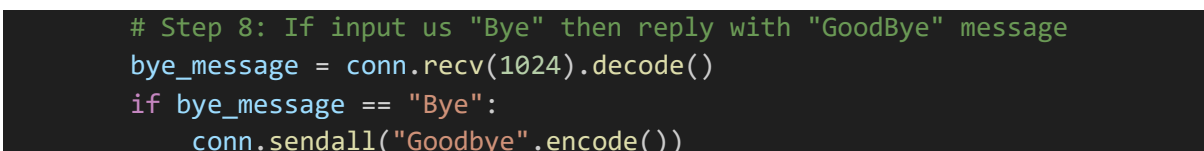
OUTPUT in the client.py terminal :



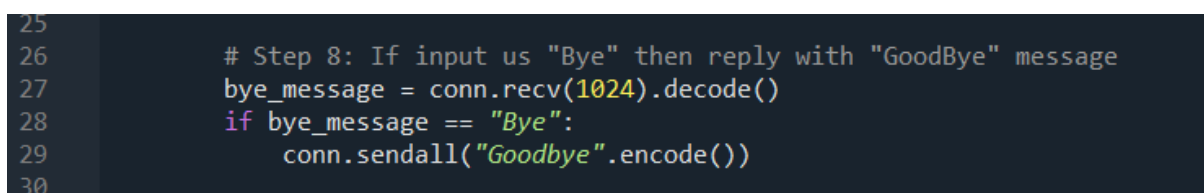
```
In [1]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
```

8. If the user has typed "Bye", the server must reply with "Goodbye".

server.py

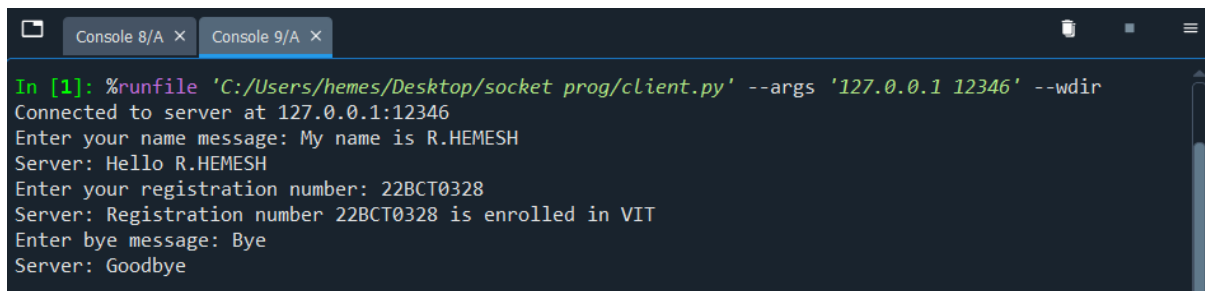


```
# Step 8: If input us "Bye" then reply with "GoodBye" message
bye_message = conn.recv(1024).decode()
if bye_message == "Bye":
    conn.sendall("Goodbye".encode())
```



```
25
26     # Step 8: If input us "Bye" then reply with "GoodBye" message
27     bye_message = conn.recv(1024).decode()
28     if bye_message == "Bye":
29         conn.sendall("Goodbye".encode())
30
```

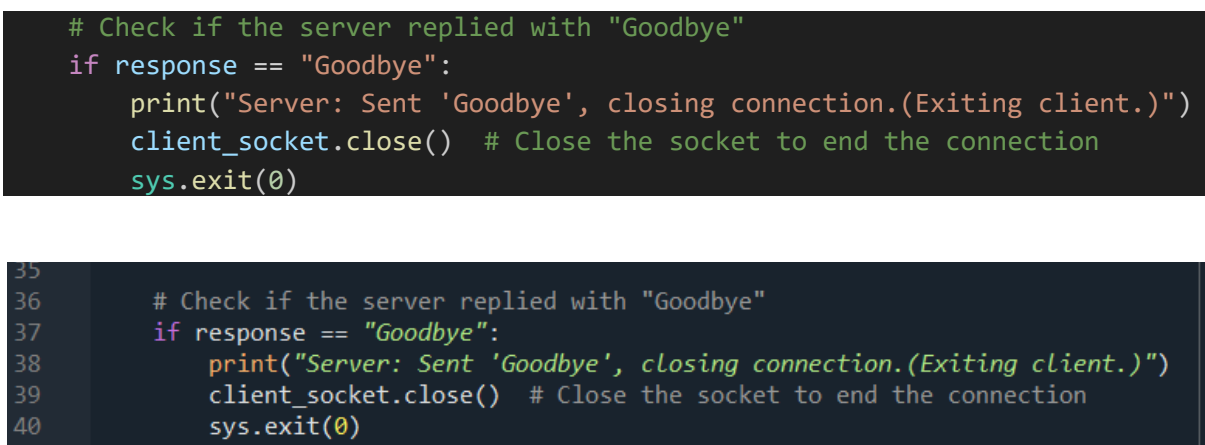
OUTPUT in the client.py terminal :



```
In [1]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
Server: Goodbye
```

9. If the server replied with a "Goodbye", the client quits.

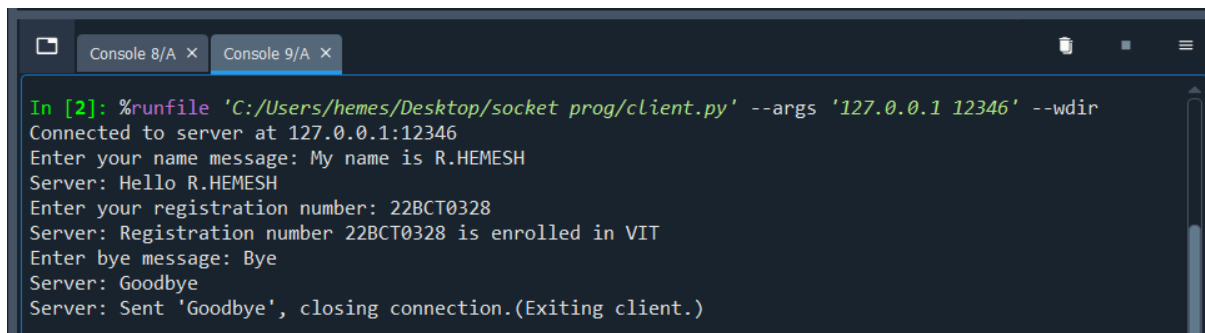
client.py:



```
# Check if the server replied with "Goodbye"
if response == "Goodbye":
    print("Server: Sent 'Goodbye', closing connection.(Exiting client.)")
    client_socket.close() # Close the socket to end the connection
    sys.exit(0)
```

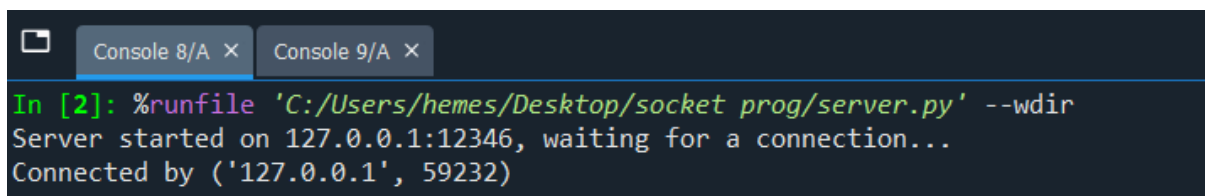
```
35
36     # Check if the server replied with "Goodbye"
37     if response == "Goodbye":
38         print("Server: Sent 'Goodbye', closing connection.(Exiting client.)")
39         client_socket.close() # Close the socket to end the connection
40         sys.exit(0)
```

OUTPUT in the client.py terminal :



```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
Server: Goodbye
Server: Sent 'Goodbye', closing connection.(Exiting client.)
```

OUTPUT in the server.py terminal:



```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/server.py' --wdir
Server started on 127.0.0.1:12346, waiting for a connection...
Connected by ('127.0.0.1', 59232)
```

FULL CODE FOR ALL 9 STEPS (server.py, client.py, output) :

```
1 import socket
2
3 HOST = '127.0.0.1'
4 PORT = 12346
5 with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
6     server_socket.bind((HOST, PORT))
7     server_socket.listen()
8     print(f"Server started on {HOST}:{PORT}, waiting for a connection...")
9
10 # Wait for a client connection
11 conn, addr = server_socket.accept()
12 with conn:
13     print(f"Connected by {addr}")
14
15     # Step 4: Receive and respond to the name message
16     name_message = conn.recv(1024).decode()
17     if name_message.startswith("My name is"):
18         name = name_message.split("My name is ")[1]
19         conn.sendall(f"Hello {name}".encode())
20
21     # Step 6: Receive and respond to the registration number
22     reg_number = conn.recv(1024).decode()
23     conn.sendall(f"Registration number {reg_number} is enrolled in VIT".encode())
24
25     # Step 8: If input us "Bye" then reply with "GoodBye" message
26     bye_message = conn.recv(1024).decode()
27     if bye_message == "Bye":
28         conn.sendall("Goodbye".encode())
29
```

```
1 import socket
2 import sys
3 # Check if IP and port are provided on the command line
4 if len(sys.argv) != 3:
5     print("Usage: client.py <server_ip> <port>")
6     sys.exit(1)
7
8 # Retrieve IP and port from the command-line arguments
9 server_ip = sys.argv[1]
10 server_port = int(sys.argv[2])
11
12 # Connect to the server
13 with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
14     client_socket.connect((server_ip, server_port))
15     print(f"Connected to server at {server_ip}:{server_port}")
16
17 # Step 3: Ask for the user's name
18 name_message = input("Enter your name message: ")
19 client_socket.sendall(name_message.encode())
20 response = client_socket.recv(1024).decode()
21 print("Server:", response)
22
23 # Step 5: Send registration number to the server
24 reg_number = input("Enter your registration number: ")
25 client_socket.sendall(reg_number.encode())
26 response = client_socket.recv(1024).decode()
27 print("Server:", response)
28
29 # Step 7: Send "Bye" message to the server
30 bye_msg=input("Enter bye message: ")
31 client_socket.sendall(bye_msg.encode())
32 response = client_socket.recv(1024).decode()
33 print("Server:", response)
34
35 # Check if the server replied with "Goodbye"
36 if response == "Goodbye":
37     print("Server: Sent 'Goodbye', closing connection.(Exiting client.)")
38     client_socket.close() # Close the socket to end the connection
39     sys.exit(0)
```

```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346'
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
Server: Goodbye
Server: Sent 'Goodbye', closing connection.(Exiting client.)
```

Server.py

```
import socket

HOST = '127.0.0.1'
PORT = 12346
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as server_socket:
    server_socket.bind((HOST, PORT))
    server_socket.listen()
    print(f"Server started on {HOST}:{PORT}, waiting for a connection...")

    # Wait for a client connection
    conn, addr = server_socket.accept()
    with conn:
        print(f"Connected by {addr}")

        # Step 4: Receive and respond to the name message
        name_message = conn.recv(1024).decode()
```



```

        if name_message.startswith("My name is"):
            name = name_message.split("My name is ")[1]
            conn.sendall(f"Hello {name}".encode())

        # Step 6: Receive and respond to the registration number
        reg_number = conn.recv(1024).decode()
        conn.sendall(f"Registration number {reg_number} is enrolled in VIT".encode())

        # Step 8: If input us "Bye" then reply with "GoodBye" message
        bye_message = conn.recv(1024).decode()
        if bye_message == "Bye":
            conn.sendall("Goodbye".encode())

```

client.py

```

import socket
import sys
# Check if IP and port are provided on the command line
if len(sys.argv) != 3:
    print("Usage: client.py <server_ip> <port>")
    sys.exit(1)

# Retrieve IP and port from the command-line arguments
server_ip = sys.argv[1]
server_port = int(sys.argv[2])
# Connect to the server
with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as client_socket:
    client_socket.connect((server_ip, server_port))
    print(f"Connected to server at {server_ip}:{server_port}")

    # Step 3: Ask for the user's name
    name_message = input("Enter your name message: ")
    client_socket.sendall(name_message.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)

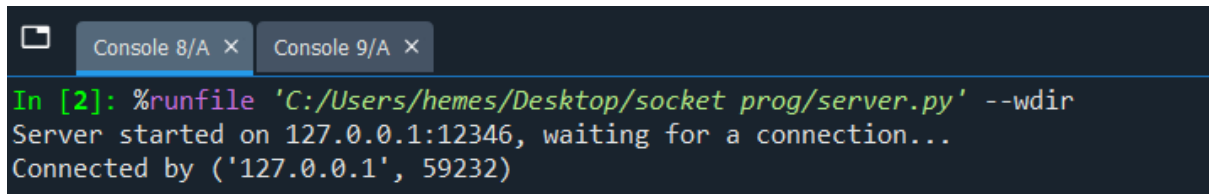
    # Step 5: Send registration number to the server
    reg_number = input("Enter your registration number: ")
    client_socket.sendall(reg_number.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)

    # Step 7: Send "Bye" message to the server
    bye_msg=input("Enter bye message: ")
    client_socket.sendall(bye_msg.encode())
    response = client_socket.recv(1024).decode()
    print("Server:", response)

```

```
# Check if the server replied with "Goodbye"
if response == "Goodbye":
    print("Server: Sent 'Goodbye', closing connection.(Exiting client.)")
    client_socket.close() # Close the socket to end the connection
    sys.exit(0)
```

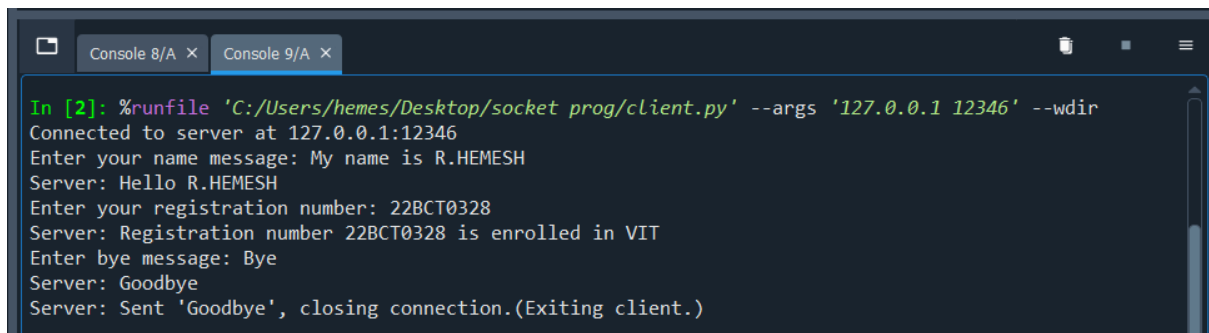
OUTPUT in the server.py terminal :



The screenshot shows a Jupyter Notebook interface with two console tabs: 'Console 8/A' and 'Console 9/A'. The active tab is 'Console 9/A', which displays the following output:

```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/server.py' --wdir
Server started on 127.0.0.1:12346, waiting for a connection...
Connected by ('127.0.0.1', 59232)
```

OUTPUT in the client.py terminal:



The screenshot shows a Jupyter Notebook interface with two console tabs: 'Console 8/A' and 'Console 9/A'. The active tab is 'Console 9/A', which displays the following output:

```
In [2]: %runfile 'C:/Users/hemes/Desktop/socket prog/client.py' --args '127.0.0.1 12346' --wdir
Connected to server at 127.0.0.1:12346
Enter your name message: My name is R.HEMESH
Server: Hello R.HEMESH
Enter your registration number: 22BCT0328
Server: Registration number 22BCT0328 is enrolled in VIT
Enter bye message: Bye
Server: Goodbye
Server: Sent 'Goodbye', closing connection.(Exiting client.)
```