Muhammad Hassaan Raza BSCS 2022 FAST NU LHR (22L-6947)

Computer Networks
Lab Manual 5
Server Code

```
import socket



def initDatabase():
   # data for 10 students using a simple 2D array
   global studentDatabase
   global presentStudentList
   presentStudentList = []
   studentDatabase = [
       ['22-0000', 0],
       ['22-0001', 0],
       ['22-0002', 0],
       ['22-0003', 0],
       ['22-0004', 0],
       ['22-0005', 0],
       ['22-0006', 0],
       ['22-0007', 0],
       ['22-0008', 0],
       ['22-0009', 0]
   ]



def toggleAttendance(msg):
   global studentDatabase
   rollNum=msg[:-3]
   attendance=msg[-2:].upper()
   studentFound=False
   i=1
   for elem in studentDatabase:
       if (elem[0]==rollNum):
           studentFound=True
           if (elem[1]==0):
               if (attendance=='CI'):
                   print("Welcome Student ", rollNum)
                   presentStudentList.append(rollNum)
                   elem[1]=1
               elif (attendance=='CO'):
```

```
                    print("You didn't check in today, contact system administrator.")
                else:
                    print("Incorrect comand!")
                    studentFound=False
            elif (elem[1]==1):
                if (attendance=='CI'):
                    print("You are already here.")
                elif (attendance=='CO'):
                    print(f"GoodBye Student {rollNum}, have a nice day!")
                    presentStudentList.pop(presentStudentList.index(rollNum))
                    elem[1]=0
                else:
                    print("Incorrect comand!")
                    studentFound=False
            break
    if (studentFound):
        print("The students present are: ")
        for val in presentStudentList:
            print(f"{i}: {val}")
            i+=1
    else:
        print("Roll number not found!")



def createSocket():
    # Create a UDP socket
    try:
        global sock
        # As we are using UDP evident from the second arguement so don't need to listen
i.e. sock.listen()
        # Note that a UDP packet is called a datagram
        sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
    except socket.error as msg:
        print("Socket Creation Error: ", msg)

def bindSocket():
        try:
            # Bind the socket to the server address and port
            server_address = ('127.0.0.1', 2000)    #(host, port)->tuple; note that
this IP is for localhost
            sock.bind(server_address)
```

```
        except socket.error as msg:
            print("Socket Binding Error: ", msg)

def main():
    initDatabase()
    createSocket()
    bindSocket()

    print("Socket created and bound")
    print("Listening for messages...\n")

    sock.settimeout(20)  #Invokes the socket timeout exveption if no send/receive
within 20 seconds

    contListening=True
    while contListening:
        try:
            # Receive the message from the client
            client_message, client_address = sock.recvfrom(2000)    #receive the data
and address pair
            # print(f"Received message from IP: {client_address[0]} and Port No:
{client_address[1]}")
            # print(f"Client Message: {client_message.decode()}")

            # Send the message back to the client
            sock.sendto(client_message, client_address)

            # marks/unmarks the attendance for the student
            toggleAttendance(client_message.decode())

            if client_message.decode().strip().lower()=="exit":
                contListening=False

        except socket.error as msg:
            print("Socket timeout: ", msg)
            break

        except Exception as e:
            print(f"An error occurred: {e}")
            break
    # Closing the socket
    print("Server Closing...\n")
```

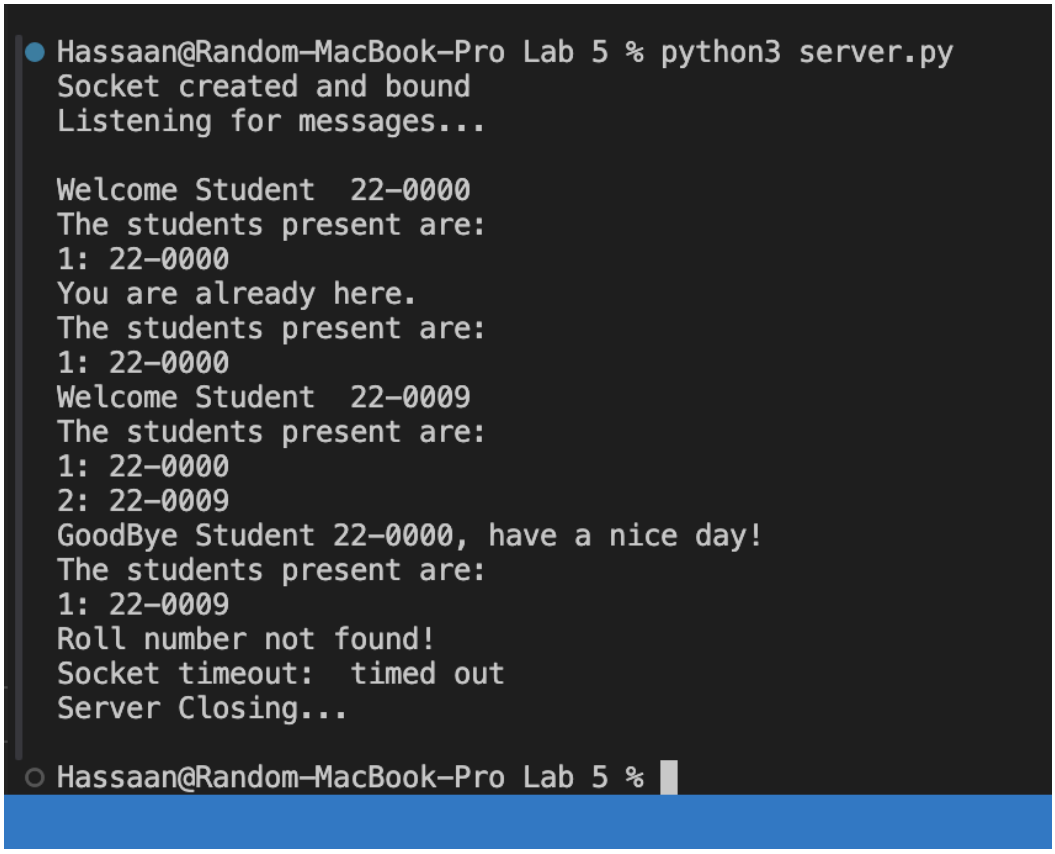
```
    sock.close()

if __name__ == "__main__":
    main()
```

Client Code

```
import socket

def main():
    # Create a UDP socket
    sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)

    server_address = ('127.0.0.1', 2000)

    try:
        # Get input from the user
        client_message = input("Enter Message: ")

        # Send the message to the server
        sock.sendto(client_message.encode(), server_address)

        # Receive the response from the server
        server_message, _ = sock.recvfrom(2000)
        print(f"Server Message: {server_message.decode()}")

    except Exception as e:
        print(f"An error occurred: {e}")

    finally:
        # Close the socket
        sock.close()

if __name__ == "__main__":
    main()
```

Output Screenshots

Server Side:



Client Side: