**DS Lab 4**

**Name:** GP Anirudh

**Registration Number:** 180905452

**Section:** B

**Batch:** B2

**Roll Number:** 59

**Practice Programs:**

**1. Write a program where client can send a message to the server and the server can receive the message and send, or echo, it back to the client.**

**Echo Client:**

import socket  
HOST = '127.0.0.1' # The server's hostname or IP address  
PORT = 2053 # The port used by the server

with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as s:

s.connect((HOST, PORT))

s.sendall(b'Hello, world')

data = s.recv(1024)

print('Received Connection')

print('Server:', data.decode())

**Echo Server:**

import socket  
HOST = '127.0.0.1' # Standard loopback interface address (localhost)

PORT = 2053

# Port to listen on (non-privileged ports are > 1023)  
with socket.socket(socket.AF\_INET, socket.SOCK\_STREAM) as s:

s.bind((HOST, PORT))

s.listen()  
 conn, addr = s.accept()

with conn:  
 print('Connected by', addr)

while True:

data = conn.recv(1024)

if data:

print("Client: ",data.decode())  
 data = input("Enter message to client:");

if not data:

break;  
# sending message as bytes to client.

conn.sendall(bytearray(data, 'utf-8'));

conn.close()

**Output:**

**Text

Description automatically generated**

**Text

Description automatically generated**

**2. Write a program to create TCP time server in Python**

**Time Client:**

import socket

# create a socket object  
s = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)   
host = socket.gethostname()  
port = 9991

# connection to hostname on the port.  
s.connect((host, port))

# Receive no more than 1024 bytes  
tm = s.recv(1024)  
print(' Current time from Sever :', tm.decode())  
s.close()

**Time Server:**

import socket  
import time

# create a socket object  
serversocket = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

# get local machine name

host = socket.gethostname() port = 9991

# bind to the port serversocket.bind((host, port)) # queue up to 5 requests serversocket.listen(5)

while True:  
 # establish a connection

clientsocket,addr = serversocket.accept()

print("Got a connection from %s" % str(addr))

currentTime = time.ctime(time.time()) + "\r\n" clientsocket.send(currentTime.encode('ascii'))

clientsocket.close()

**Output:**

**Text

Description automatically generated**

**Text

Description automatically generated**

**3.) Write a TCP chat server in python using socket programming.**

**Client Chat:**

import socket  
HOST = '172.16.58.193' # Standard loopback interface address (localhost) PORT = 31621 # Port to listen on (non-privileged ports are > 1023)  
s = socket.socket()  
name = input(str("\nEnter your name: "))  
print("\nTrying to connect to ", HOST, "(", PORT, ")\n")

s.connect((HOST, PORT))  
print("Connected...\n")  
s.send(name.encode())  
s\_name = s.recv(1024)  
s\_name = s\_name.decode()  
print(s\_name, "has joined the chat room\nEnter [e] to exit chat room\n")

while True:

message = s.recv(1024)

message = message.decode()

print(s\_name, ":", message)

message = input(str("Me : "))

if message == "[e]":

message = "Left chat room!" s.send(message.encode()) print("\n")  
 break

s.send(message.encode())

**Server Chat:**

# server.py  
import socket  
HOST = '127.0.0.1' # Standard loopback interface address (localhost)  
PORT = 31621 # Port to listen on (non-privileged ports are > 1023)  
s = socket.socket()  
s.bind((HOST, PORT))  
s.listen()  
print("\nWaiting for incoming connections...\n")  
conn, addr = s.accept()  
print("Received connection from ", addr[0], "(", addr[1], ")\n")  
s\_name = conn.recv(1024)  
s\_name = s\_name.decode()  
print(s\_name, "has connected to the chat room\nEnter [e] to exit chat room\n") name = input(str("Enter your name: "))  
conn.send(name.encode())

while True:

message = input(str("Me : "))

if message == "[e]":

message = "Left chat room!"

conn.send(message.encode())

print("\n")  
 break

conn.send(message.encode())

message = conn.recv(1024)

message = message.decode()

print(s\_name, ":", message)

**Output:**

**Text

Description automatically generated**

**Text

Description automatically generated**

**4A. Forking/ Threading (Concurrent Server)**

**Client :**

import socket  
ClientSocket = socket.socket() host = '127.0.0.1'  
port = 11596  
print('Waiting for connection')

try:

ClientSocket.connect((host, port))

except socket.error as e:

print(str(e))  
Response = ClientSocket.recv(1024)

while True:

Input = input('Client Say Something: ') ClientSocket.send(str.encode(Input))

Response = ClientSocket.recv(1024)

print('From Server : ' + Response.decode())

ClientSocket.close()

**Server:**

import socket  
import os  
from \_thread import \*

ServerSocket = socket.socket()

host = '127.0.0.1'  
port = 11596  
ThreadCount = 0

try:

ServerSocket.bind((host, port))

except socket.error as e:

print(str(e))

print('Waitiing for a Connection..')

ServerSocket.listen(5)

def threaded\_client(connection):

connection.send(str.encode('Welcome to the Server'))

while True:

data = connection.recv(2048)  
 print('Received from client :' + str(ThreadCount) +data.decode()) Inputs = input('Server Says: ')

if not data:

break

connection.sendall(Inputs.encode())

connection.close()

while True:

Client, address = ServerSocket.accept()

print('Connected to: ' + address[0] + ':' + str(address[1])) start\_new\_thread(threaded\_client, (Client, ))

ThreadCount += 1  
 print('Thread Number: ' + str(ThreadCount))

ServerSocket.close()

**Output:**

**Text

Description automatically generated**

**Text

Description automatically generated**

**Lab Exercise:**

**1. Write a UDP time server to display the current time and day.**

**#client.py**

import socket  
# create a socket object  
s = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

# get local machine name  
udp\_host = socket.gethostname()  
udp\_port = 9991

#Send some message so the server gets client address  
msg = 'hi'  
s.sendto(msg.encode(),(udp\_host,udp\_port))

# Receive no more than 1024 bytes  
tm = s.recv(1024)  
print(' Current time from Sever :', tm.decode())  
s.close()

**# server.py**

import socket  
import time

# create a socket object  
serversocket = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

# get local machine name  
host = socket.gethostname()  
port = 9991

# bind to the port  
serversocket.bind((host, port))

while True:  
 # receive a udp message

data,addr = serversocket.recvfrom(1024) print("Got a connection from %s" % str(addr))

currentTime = time.ctime(time.time()) + "\r\n"

serversocket.sendto(currentTime.encode('ascii') , addr)

**Output:**

**Text

Description automatically generated**

**Text

Description automatically generated**

**2. Write a UDP simple chat program for message send and receive.**

**#client code :**

import socket

HOST = socket.gethostname()  
PORT = 31621 # Port to listen on (non-privileged ports are > 1023)

s = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)

name = input(str("\nEnter your name: "))  
print("\nTrying to connect to ", HOST, "(", PORT, ")\n")  
s.sendto(b"" , (HOST , PORT))  
print("Connected...\n")

s.sendto(name.encode('ascii') , (HOST , PORT))  
s\_name = s.recv(1024)  
s\_name = s\_name.decode()  
print(s\_name, "has joined the chat room\nEnter [e] to exit chat room\n")

while True:

message, addr = s.recvfrom(1024) message = message.decode() print(s\_name, ":", message) message = input(str("Me : "))

if message == "[e]":  
 message = "Left chat room!" s.sendto(message.encode() , (HOST , PORT)) print("\n")  
 break

s.sendto(message.encode() , (HOST , PORT))

**#servercode :**

import socket

HOST = socket.gethostname()  
PORT = 31621 # Port to listen on (non-privileged ports are > 1023) s = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM) s.bind((HOST, PORT))

print("\nWaiting for incoming connections...\n")  
data,addr = s.recvfrom(1024)  
print("Received connection from ", addr[0], "(", addr[1], ")\n")

s\_name, addr = s.recvfrom(1024)

s\_name = s\_name.decode()

print(s\_name, "has connected to the chat room\nEnter [e] to exit chat room\n")

name = input(str("Enter your name: "))  
s.sendto(name.encode('ascii') , addr)

while True:

message = input(str("Me : ")) if message == "[e]":

message = "Left chat room!" s.sendto(message.encode() , addr) print("\n")  
break

s.sendto(message.encode() , addr) message, addr = s.recvfrom(1024) message = message.decode() print(s\_name, ":", message)

**Output:**

**Text

Description automatically generated**

**A screenshot of a computer

Description automatically generated with medium confidence**

**3. Write a TCP/UDP peer to peer chat system between two different machines.**

**#client-code:**

import socket  
HOST = '172.16.58.111'  
PORT = 31621

# Port to listen on (non-privileged ports are > 1023)  
s = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM)  
name = input(str("\nEnter your name: "))  
print("\nTrying to connect to ", HOST, "(", PORT, ")\n")  
s.sendto(b"" , (HOST , PORT))  
print("Connected...\n")  
s.sendto(name.encode('ascii') , (HOST , PORT))  
s\_name = s.recv(1024)  
s\_name = s\_name.decode()  
print(s\_name, "has joined the chat room\nEnter [e] to exit chat room\n")

while True:

message, addr = s.recvfrom(1024)

message = message.decode()

print(s\_name, ":", message)

message = input(str("Me : "))

if message == "[e]":  
 message = "Left chat room!"

s.sendto(message.encode() , (HOST , PORT))

print("\n")  
 break

s.sendto(message.encode() , (HOST , PORT))

**#server-code:**

import socket  
HOST = socket.gethostname()  
PORT = 31621 # Port to listen on (non-privileged ports are > 1023)

s = socket.socket(socket.AF\_INET, socket.SOCK\_DGRAM) s.bind((HOST, PORT))

print("\nWaiting for incoming connections...\n")  
data,addr = s.recvfrom(1024)  
print("Received connection from ", addr[0], "(", addr[1], ")\n") s\_name, addr = s.recvfrom(1024)

s\_name = s\_name.decode()  
print(s\_name, "has connected to the chat room\nEnter [e] to exit chat room\n")

name = input(str("Enter your name: "))  
s.sendto(name.encode('ascii') , addr)

while True:

message = input(str("Me : "))

if message == "[e]":

message = "Left chat room!"

s.sendto(message.encode() , addr) print("\n")  
break

s.sendto(message.encode() , addr)

message, addr = s.recvfrom(1024) message = message.decode() print(s\_name, ":", message)

**Output:**

Text

Description automatically generated

Text

Description automatically generated