CLIENT SIDE:

import socket

import threading

class Client(object):

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

def \_\_init\_\_(self):

self.s.connect(("127.0.0.1",6000))

iThread = threading.Thread(target=self.sendMsg)

iThread.daemon = True

iThread.start() #the sendMsg will run in the background

while(True):

data = self.s.recv(1024)

if not data:

break

print(str(data,"utf-8"))

def sendMsg(self):

print("Group Chat")

while True:

self.s.send(bytes(input(""), 'utf-8'))

client = Client()

SERVER SIDE:

import socket

import threading

class Server(object):

# create a sockets

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

# Store the connected clients

connections = []

def \_\_init\_\_(self):

self.s.bind(("127.0.0.1",6000))

self.s.listen(1)

def handler(self, c, addr):

while True:

data = c.recv(1024)

# print("count {},".format(threading.active\_count()))

for conn in self.connections:

if conn != c:

conn.send(data)

if not data:

print("{}:{} Disconnected".format(addr[0],addr[1]))

self.connections.remove(c)

c.close()

break

# print(data)

def run(self):

print("Server started ....")

while True:

c, addr = self.s.accept()

cThread = threading.Thread(target=self.handler, args=(c, addr))

cThread.daemon = True #kill the thread when the main process exits

cThread.start()

self.connections.append(c)

print("{}:{} Connected".format(addr[0], addr[1]))

server = Server()

server.run()

OUTPUT:

