**19CSE301- COMPUTER NETWORK**

**Case Study**

**Online Movie ticket booking system**

**Group Number - 16**

|  |  |  |  |
| --- | --- | --- | --- |
| **Registration No** | **Name** | **Email ID** | **Contribution** |
| CB.EN.U4CSE19020 | G. ASHISH | cb.en.u4cse19020@cb.students.amrita.edu | Reviews & Rating |
| CB.EN.U4CSE19037 | N. SATHWIK | cb.en.u4cse19037@cb.students.amrita.edu | Ticket booking |
| CB.EN.U4CSE19047 | R.VENKATESH PRASAD | cb.en.u4cse19047@cb.students.amrita.edu | Upcoming movies |

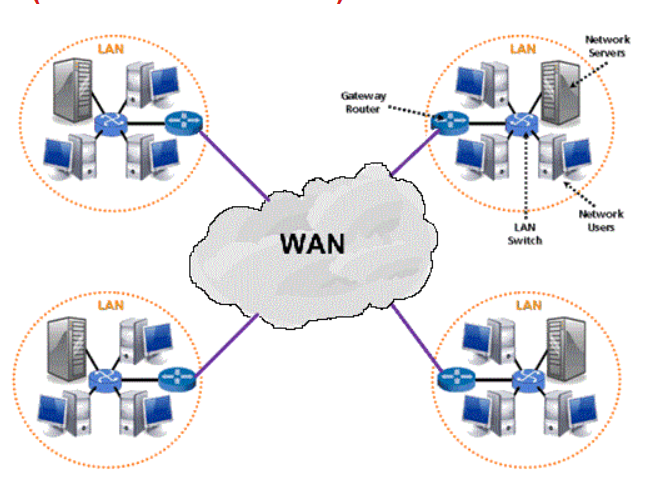
**Type of network:**

WAN and VLAN

WAN(wide area network):

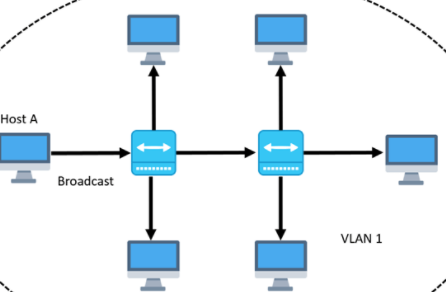
A wide-area (WAN) network is a communication network that typically covers a large geographical area, primarily used for connecting computers. Cities, states, or even countries can be connected with WANs.

WANs are commonly used by larger businesses or organizations and are used by businesses in a broader variety of sectors with networks in many places.



**VLAN(virtual local area network):**

A VLAN (virtual LAN) is a subnetwork which can group together collections of devices on separate physical local area networks (LANs). A LAN is a group of computers and devices that share a communications line or wireless link to a server within the same geographical area.



**2) Client Configuration :**

64-bit Microsoft® Windows® 8/10

x86\_64 CPU architecture; 2nd generation Intel Core or newer, or AMD CPU with support for a Windows Hypervisor

8 GB RAM or more

8 GB of available disk space minimum

1280 x 800 minimum screen resolution

Server requirements

Processor: AMD EPYC 7003 (64 Cores and 128 Threads)

RAM: 1TB 2500Hz

Storage: 100 TB SSD Operating systems:

Windows Server 20H2 version 10.0.19042

Graphics Card: NVDIA Tesla V100S (640 Tensor cores and 5210 CUDO Cores)

**3.) Server Configuration:**

Processor: AMD EPYC 7003 (64 Cores and 128 Threads)

RAM: 1TB 2500Hz

Storage: 100 TB SSD Operating systems:

Windows Server 20H2 version 10.0.19042

Graphics Card: NVDIA Tesla V100S (640 Tensor cores and 5210 CUDO Cores)

**4.) Servers Used :**

a) client servers

b) FTP server

c) mail server

d) SQL server

**5)** **Network cable:**

**Coaxial cable** because fiber optic cable used for longer distances to transmit data in light speed but it is costly

Whereas coaxial cable Coaxial cable is used as a [transmission line](https://en.wikipedia.org/wiki/Transmission_line) for radio frequency signals we transmit signals in most of airports and in MAN’s we generally use coaxial cable and it has a data rate of 600Mbps

**Type of Topology**: **Star Topology**

**Different Departments Within The Application:**

**1) Movie ticket booking Department:**

When the client requests the server for a ticket to a specific movie, the server provides the list of movies which are available to watch on that day. It is the client’s wish to select the snacks or not. It includes 3 systems for accepting requests and 2 systems for secure payment.

**2) Upcoming Movies :**

In This Department, on the client’s request, he/she can view the upcoming movies which are yet to release. And if a client requests for a specific movie, then the server will provide the details of that specific movie to the client. It includes 2 systems.

**3) Reviews and Ratings :**

In This We Will be able view the ratings by the users so that client have an idea about the movie they wanted to watch. If the client selects particular movie all the reviews related to the movie will be displayed to the client. Here we also provided the client to give review on the movie he watched.

**Why is Networking required?**

Computer networks help users on the network to share the resources and in communication. File sharing: Networking of computers helps the network users to share data files. Hardware sharing: Users can share devices such as printers, scanners, CD-ROM drives, hard drives etc.

**NETWORK PERFORMANCE:**

**Network latency :**

Latency can be used to measure network delays, focusing on the time spent in the successful transfer of packets or a packet of data from one point to another within a network.

Latency = Propagation + Transmit + Queue

Propagation = Distance/SpeedOfLight

Transmit = Size/Bandwidth

**Packet loss :**

Packet loss refers to the number of packets that were successfully sent out from one point in a network, but never got to their destination.

Efficiency = 100% \* (transferred - retransmitted) / transferred

Network Loss = 100 - Efficiency

**Bandwidth :**

Refers to the number of data that can be transmitted from one point to another in a network, within a given time.

BW = fc/Q Where fc = resonant frequency Q = quality factor

For ethernet general scenario is 10 Mbit/s

**Throughput:**

Throughput, on the other hand, is the number of data that got transmitted from one point to another within the given time. A network performance measurement is created when the throughput is analysed against the bandwidth.

**Throughput = TransferSize / TransferTime**

**TransferTime = RTT + 1/Bandwidth x TransferSize**

**Effective throughput 🡪 1 MB / 108 ms = 74.1 Mbps**

**Processing delay :**

**dividing the distance (the length of the medium) in meter by its propagation speed in m/s [L/R]**

**queing delay :I(L/R)(1 - I) for I <1**

**Transmission time:**

**The time required for transmission of a message depends on the size of the message and the bandwidth of the channel.**

**Transmission time=Message size / Bandwidth**

**Propagating time:**

**Propagation time measures the time required for a bit to travel from the source to the destination. The propagation time is calculated by dividing the distance by the propagation speed.**

**Propagation time = Distance /Propagation speed**

we had used four csv files they are:

1. Ticket.csv

It contains mname, genre, price, time, place, Seats, hours and leftseats .

1. Rating.csv

It contains information about the person and their reviews and ratings.

1. Upcoming.csv

It contains information about the upcoming movies.

1. Snacks.csv

It contains about the snack items

List of operations completed with the File

1. Insertion
2. View
3. Updating
4. Printing the data with of users choice
5. Extracting the specific data from the csv file

**Server-side code:**

from os import close

import socket

import re

from numpy import empty

import pandas as pd

import threading

from datetime import datetime

from pytz import timezone

format = "%H:%M"

# Current time in UTC

now\_utc = datetime.now(timezone('UTC'))

# Convert to Asia/Kolkata time zone

now\_asia = now\_utc.astimezone(timezone('Asia/Kolkata'))

time=now\_asia.strftime(format)

h=re.findall("[0-9]+",str(time))

H1=int(h[0])

H2=int(h[1])/60

total=H1+H2

print(total)

DISCONNECT\_MSG = "!DISCONNECT"

HOST = ""       # server ip

PORT = 42050   # Arbitrary non-privileged port

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

try:

    s.bind((HOST, PORT))

except socket.error as error:

    # printing the error here

    print("Oops!. Please check the error and try again " + str(error))

print ("ticket booking  Server running", HOST, PORT)

s.listen()

print ("Sending Data ....")

def  update (movie\_name,seats):

    df=pd.read\_csv("ticket.csv")

    x=df.iloc[movie\_name]["leftseats"]

    df.at[movie\_name, "leftseats"] -= seats

    df.to\_csv("ticket.csv",index=7)

    x=df.at[movie\_name, "price"]

    x=x\*int(seats)

    print(df)

    return x

def handle\_client(conn, addr):

    print(f"[NEW CONNECTION] {addr} connected.")

    connected = True

    while connected:

        msg = conn.recv(1024).decode()

        if msg == DISCONNECT\_MSG:

            connected = False

        elif(str(msg)=="A"):

            print("option 1 selected")

            df=pd.read\_csv("ticket.csv")

            df=df[df["hours"]>total][['mname','genre','price','time','place','Seats','leftseats']]

            conn.send(str(df).encode())

            if(df is not empty):

                mov=int(conn.recv(1024).decode())

                st=int(conn.recv(1024).decode())

                t=update(mov,st)

                d1=pd.read\_csv("Snacks.csv")

                conn.send(str(d1).encode())

                sitem=int(conn.recv(1024).decode())# no.of items

                df=pd.read\_csv("Snacks.csv")

                while(sitem>0):

                    i=int(conn.recv(1024).decode())#index of  items

                    x=df.at[i,"Price"]

                    j=conn.recv(1024).decode()#quantity

                    t=t+int(x)\*int(j)

                    sitem=sitem-1

                conn.send(str(t).encode())

                t=0

        elif(str(msg)=="B"):

            print("option 2 selected")

            opt=conn.recv(1024).decode()

            if(opt=="D"):

                df=pd.read\_csv("rating.csv")

                conn.send(str(df).encode())

            else:

                rname=conn.recv(1024).decode()

                rmov=conn.recv(1024).decode()

                rating=conn.recv(1024).decode()

                review=conn.recv(1024).decode()

                df=pd.read\_csv("rating.csv")

                data=[pd.Series([rname,rmov,rating,review],index=df.columns)]

                df=df.append(data,ignore\_index=True)

                df.to\_csv("rating.csv",index=False)

                print(df)

                conn.send(str("Review inserted").encode())

        elif(str(msg)=="C"):

            print("option 3 selected")

            df=pd.read\_csv("upcoming.csv")

            df=df[["mname"]]

            conn.send(str(df).encode())

            umov=conn.recv(1024).decode()

            d=pd.read\_csv("upcoming.csv")

            d.set\_index("mname", inplace = True)

            x=d.loc[umov]

            print(x)

            conn.send(str(x).encode())

        print(f"[{addr}] {msg}")

        # msg = f"Msg received: {msg}"

    conn.close()

while True:

    conn , addr =s.accept()

    thread = threading.Thread(target=handle\_client, args=(conn, addr))

    thread.start()

    print("recieved from client")

**Client-side code:**

from os import rename

import socket

from datetime import datetime

from numpy import empty

from pytz import timezone

format = "%H:%M"

# Current time in UTC

now\_utc = datetime.now(timezone('UTC'))

# Convert to Asia/Kolkata time zone

now\_asia = now\_utc.astimezone(timezone('Asia/Kolkata'))

DISCONNECT\_MSG = "!DISCONNECT"

HOST = socket.gethostbyname(socket.gethostname())      # The remote host

PORT = 42050              # The same port as used by the server

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

client.connect((HOST, PORT))

print("A.Ticket Booking\nB.Reviews & Rating\nC.Upcoming movies")

connected = True

while connected:

    msg = str(input("> "))

    if msg == DISCONNECT\_MSG:

        connected = False

    elif (str(msg)=="A"):

        client.send(str(msg).encode())

        data=client.recv(1024).decode()

        print(data)

        if(data is not empty):

            mov=input("Enter the index of movie name")

            client.send(str(mov).encode())

            no\_of\_seats=input("Enter the no.of seats:")

            client.send(str(no\_of\_seats).encode())

            choise=input("Do you need to order snaks?(y/n):")

            if(choise=="y"):

                print(client.recv(1024).decode())

                sitem=input("select no.of items")

                client.send(str(sitem).encode())

                for i in range(1,int(sitem)+1):

                    item=input("select the"+ str(i)+" item index:")

                    client.send(str(item).encode())

                    qitem=input("enter the quantity of "+ str(i)+"th item:")

                    client.send(str(qitem).encode())

                x=client.recv(1024).decode()

                print("total amount is "+x)

            else:

                client.recv(1024).decode()

    elif (str(msg)=="B"):

        client.send(str(msg).encode())

        a=input("Select one of the below:\nIn order to view select D\n In order to give review select E")

        client.send(str(a).encode())

        if(a=="D"):

            client.send(str(msg).encode())

            data=client.recv(1024).decode()

            print(data)

        else:

            rname=input("Enter your name:\n")

            client.send(str(rname).encode())

            rmov=input("Enter the movie name:\n")

            client.send(str(rmov).encode())

            rating=input("Provide your rating out of 5:\n")

            client.send(str(rating).encode())

            review=input("Describe in a one word about the movie:\n")

            client.send(str(review).encode())

            print(client.recv(1024).decode())

    elif (str(msg)=="C"):

        client.send(str(msg).encode())

        print(client.recv(1024).decode())

        umov=input("Select the movie name: ")

        client.send(str(umov).encode())

        result=client.recv(1024).decode()

        print(result)

    else:

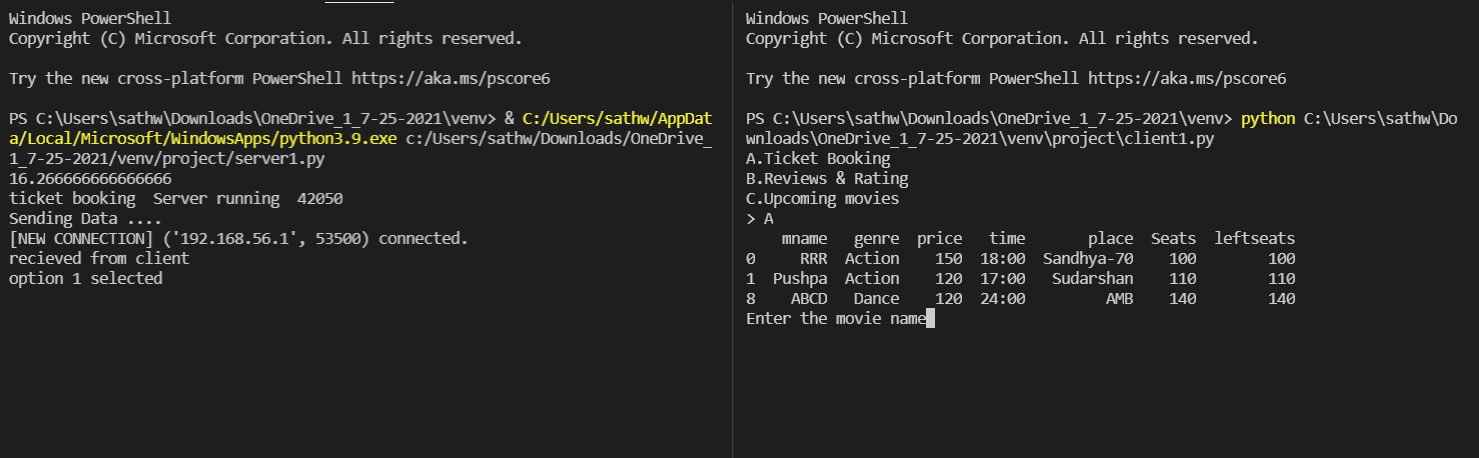
        msg = client.recv(1024).decode()

        print(f"[SERVER] {msg}")

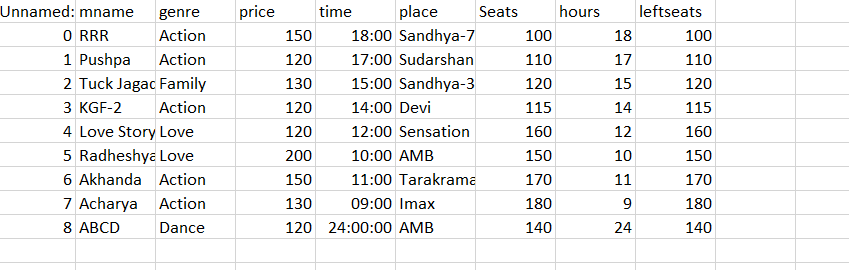
**Sample output photos:**

After selecting the ticket booking option

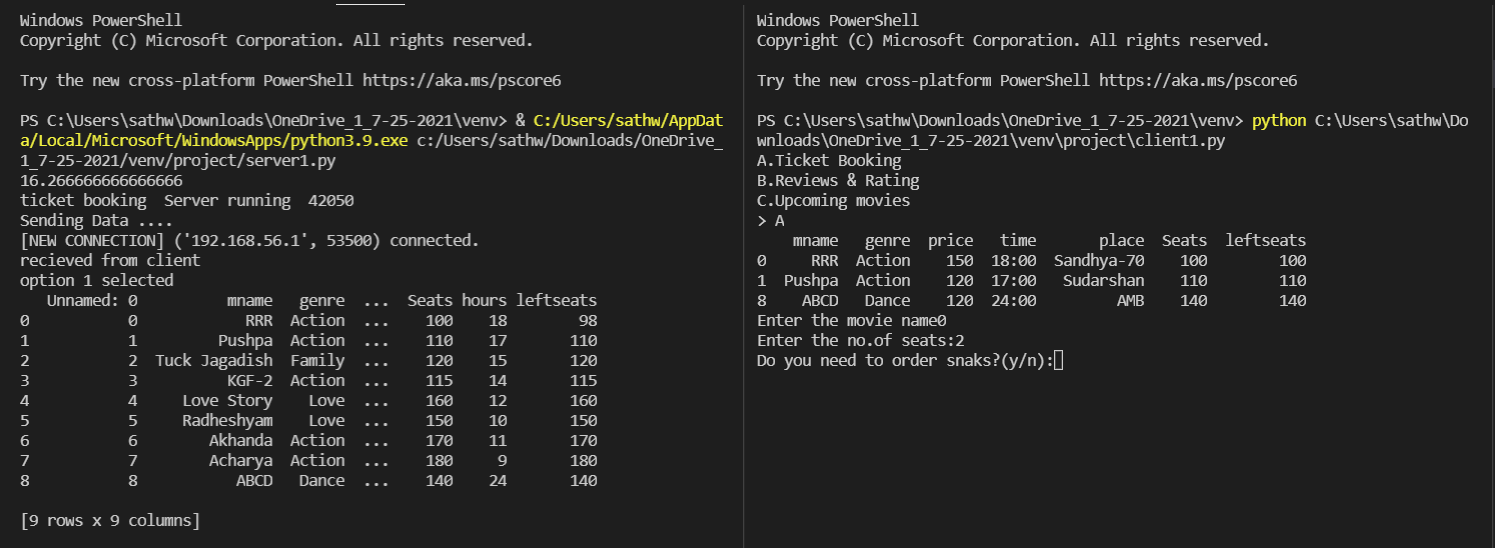
The server sends the movies which are available for the client with respect to the client’s time.

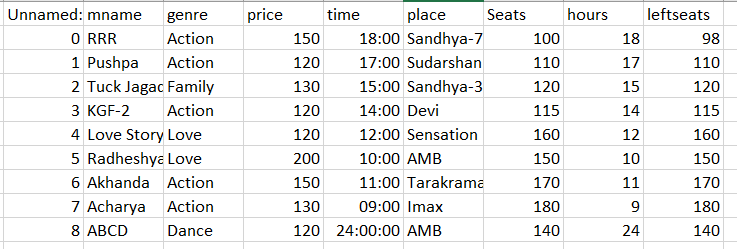


Before selecting the number of seats

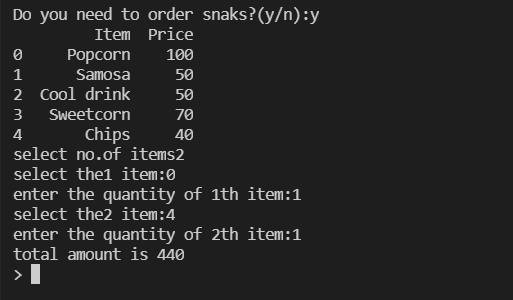


After selecting the movie index and number of seats

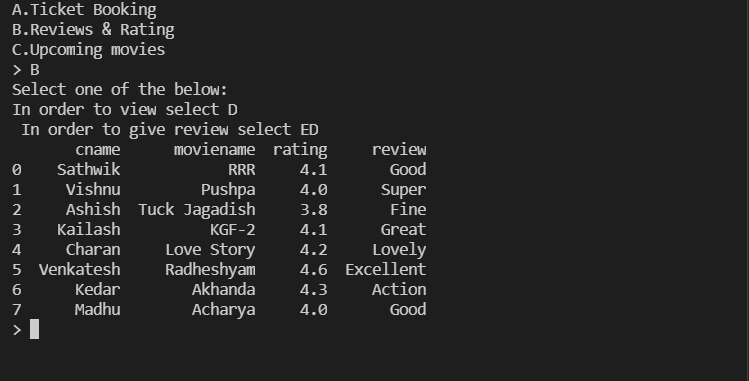




After selecting the snacks server sends the total amount that the client needed to pay

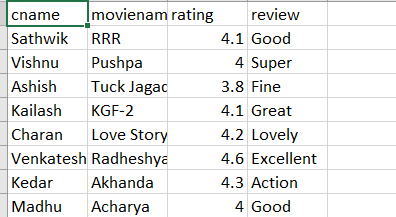


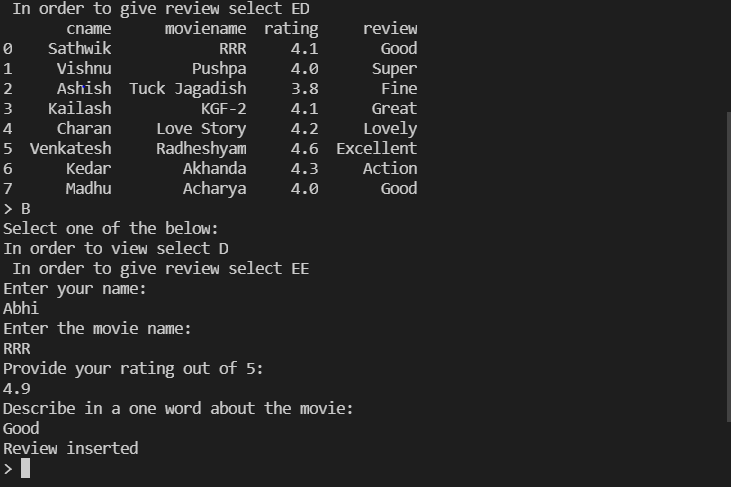
Selecting the view under review and rating option

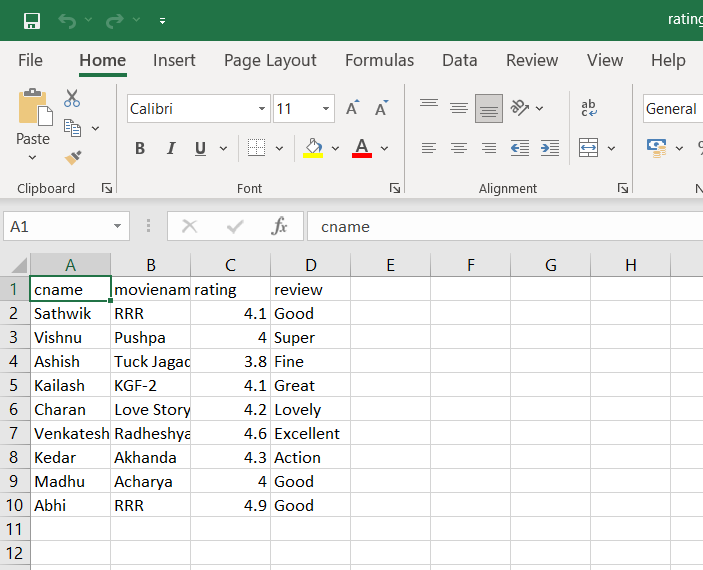


Providing review for the movie:

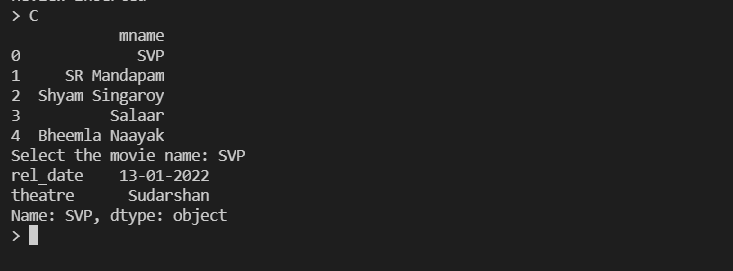
Before providing the review and rating:



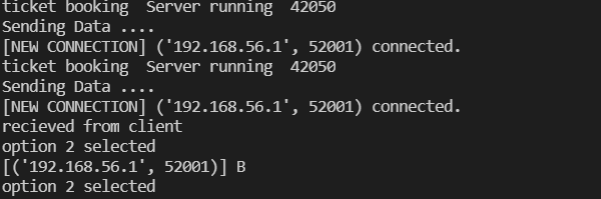


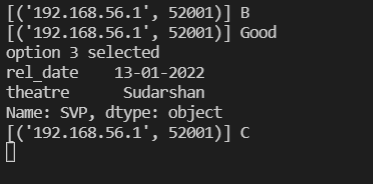
After providing the review and rating:

After selecting the option of upcoming movies:

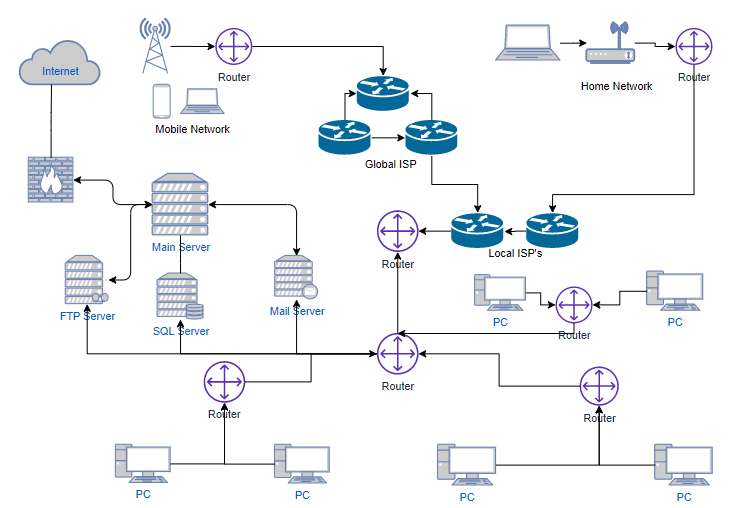


Server side output:





**ARCITECTURAL DIAGRAM:**

****