# ADVANCE JAVA DEVELOPMENT

PRACTICAL LIST

NAME: Aditi Kulkarni

ENROLLMENT NO: 101CTBMCS2122018

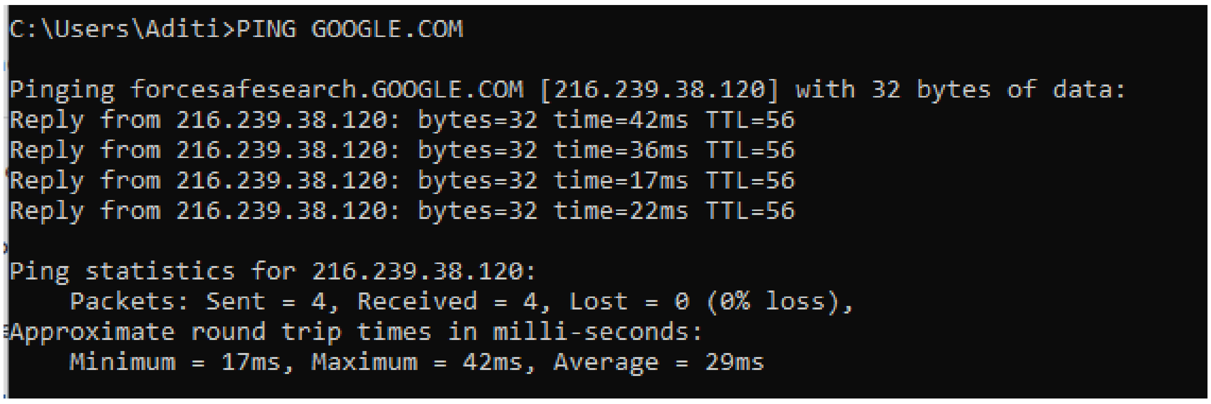
## Index

## L.1 See and study our network architecture by using various network commands such as ping,traceroute, tracert, nslookp, ipconfig,ifconfig, pathping, netstat, arp, who, whois, ncat, nmap . And understand and explain the output of each network commands. Most commands are on Ubuntu so create virtual machine and install Ubuntu in windows or install dual boot and run and see network commands on both windows and Ubuntu

### PING

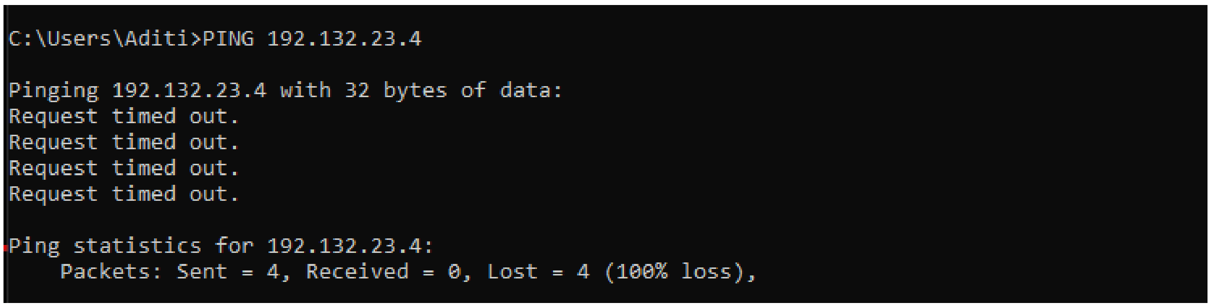
Ping command is used to test the connection between the local machine and the host server. It sends a signal to another device on the network to see if it is active. It uses the ICMP (Internet Control Message Protocol) to send out an echo request to destination device. So, this command is a subset of ICMP. After sending the echo request if an echo response is received then the device is active. It sends 4 packets each of 32 bytes and in output gives the round-trip time of the packet.

Here is an example where we have tested google.com. We can also write the IP address of google instead of domain name.



```
C:\Users\Aditi>PING GOOGLE.COM

Pinging forcesafesearch.GOOGLE.COM [216.239.38.120] with 32 bytes of data:
Reply from 216.239.38.120: bytes=32 time=42ms TTL=56
Reply from 216.239.38.120: bytes=32 time=36ms TTL=56
Reply from 216.239.38.120: bytes=32 time=17ms TTL=56
Reply from 216.239.38.120: bytes=32 time=22ms TTL=56

Ping statistics for 216.239.38.120:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 17ms, Maximum = 42ms, Average = 29ms
```

If a device doesn't exist or is not active then this output will come.



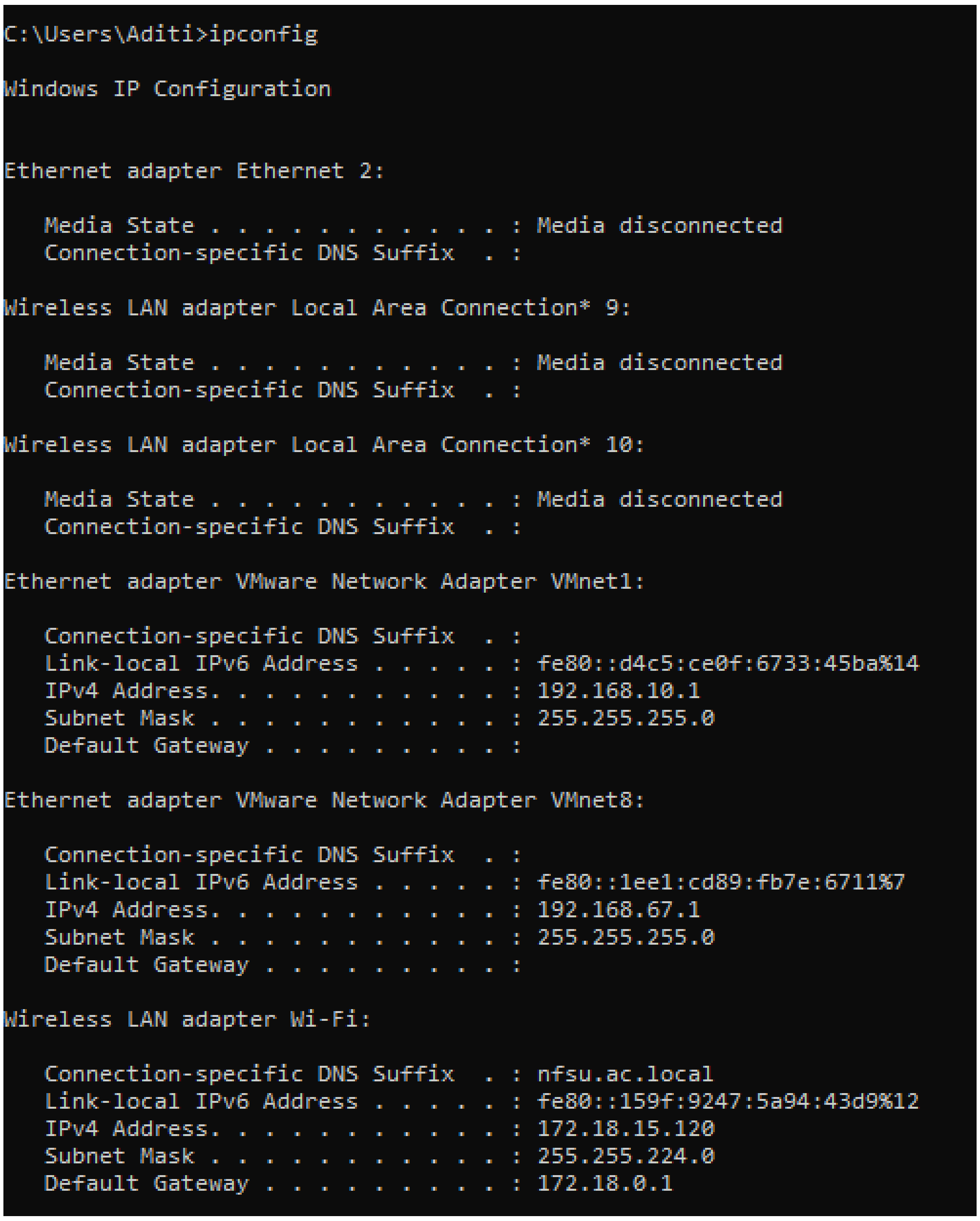
```
C:\Users\Aditi>PING 192.132.23.4

Pinging 192.132.23.4 with 32 bytes of data:
Request timed out.
Request timed out.
Request timed out.
Request timed out.

Ping statistics for 192.132.23.4:
    Packets: Sent = 4, Received = 0, Lost = 4 (100% loss),
```

Options in this command are:

- -t: Continuously pings the target until we stop it manually
- -n [count]: Specifies number of echo requests to send
- -l [size]: Sets the size of data packets
- -4 or -6: Forces use of IPv4 or IPv6

## IPCONFIG

It gives information about the IP address, subnet mask and default gateway i.e., IP address of the router for each adapter bound to TCP/IP. So, it gives information about the computer's network configuration. Ifconfig is the command used in ubuntu systems, ipconfig is used in windows.

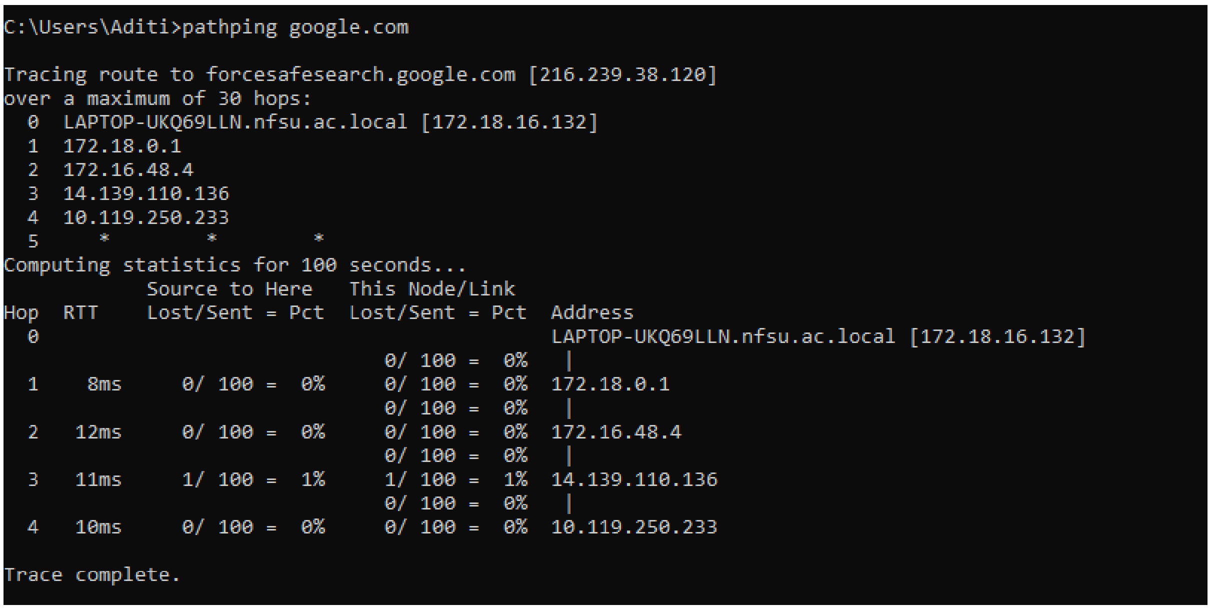


```
C:\Users\Aditi>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 9:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Wireless LAN adapter Local Area Connection* 10:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :

Ethernet adapter VMware Network Adapter VMnet1:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::d4c5:ce0f:6733:45ba%14
   IPv4 Address. . . . . . . . . . . : 192.168.10.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet8:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::1ee1:cd89:fb7e:6711%7
   IPv4 Address. . . . . . . . . . . : 192.168.67.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Wireless LAN adapter Wi-Fi:

   Connection-specific DNS Suffix  . : nfsu.ac.local
   Link-local IPv6 Address . . . . . : fe80::159f:9247:5a94:43d9%12
   IPv4 Address. . . . . . . . . . . : 172.18.15.120
   Subnet Mask . . . . . . . . . . . : 255.255.224.0
   Default Gateway . . . . . . . . . : 172.18.0.1
```

There is another add on to this command if we use *ipconfig /all* it will a detailed information about the TCP/IP configuration values of all adapters.

## PATHPING

It is the combination of traceroute and ping. Like ping, pathping sends a series of packets to the destination. It measures the round-trip time for each packet. Similar to traceroute, pathping displays the route the packets take to reach the destination. It identifies all the intermediate routers or hops along the way. For each hop, pathping conducts a series of ping measurements over a period of time, collecting statistics. It also provides insights into the quality of each hop's connection.



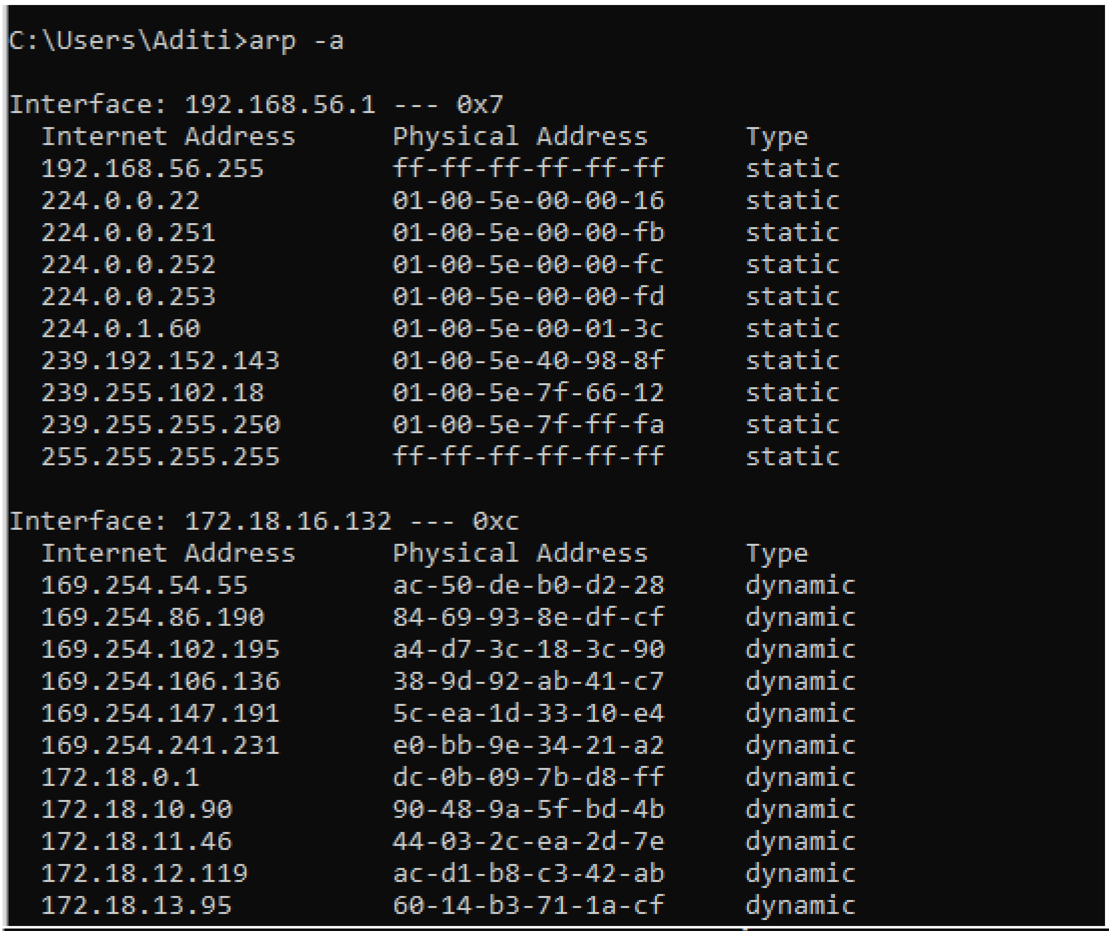
```
C:\Users\Aditi>pathping google.com

Tracing route to forcesafesearch.google.com [216.239.38.120]
over a maximum of 30 hops:
  0  LAPTOP-UKQ69LLN.nfsu.ac.local [172.18.16.132]
  1  172.18.0.1
  2  172.16.48.4
  3  14.139.110.136
  4  10.119.250.233
  5     *        *        *
Computing statistics for 100 seconds...
            Source to Here   This Node/Link
Hop  RTT    Lost/Sent = Pct  Lost/Sent = Pct  Address
  0                                           LAPTOP-UKQ69LLN.nfsu.ac.local [172.18.16.132]
                                0/ 100 =  0%   |
  1    8ms     0/ 100 =  0%     0/ 100 =  0%  172.18.0.1
                                0/ 100 =  0%   |
  2   12ms     0/ 100 =  0%     0/ 100 =  0%  172.16.48.4
                                0/ 100 =  0%   |
  3   11ms     1/ 100 =  1%     1/ 100 =  1%  14.139.110.136
                                0/ 100 =  0%   |
  4   10ms     0/ 100 =  0%     0/ 100 =  0%  10.119.250.233

Trace complete.
```

Thus, in output pathping generates a report that includes:

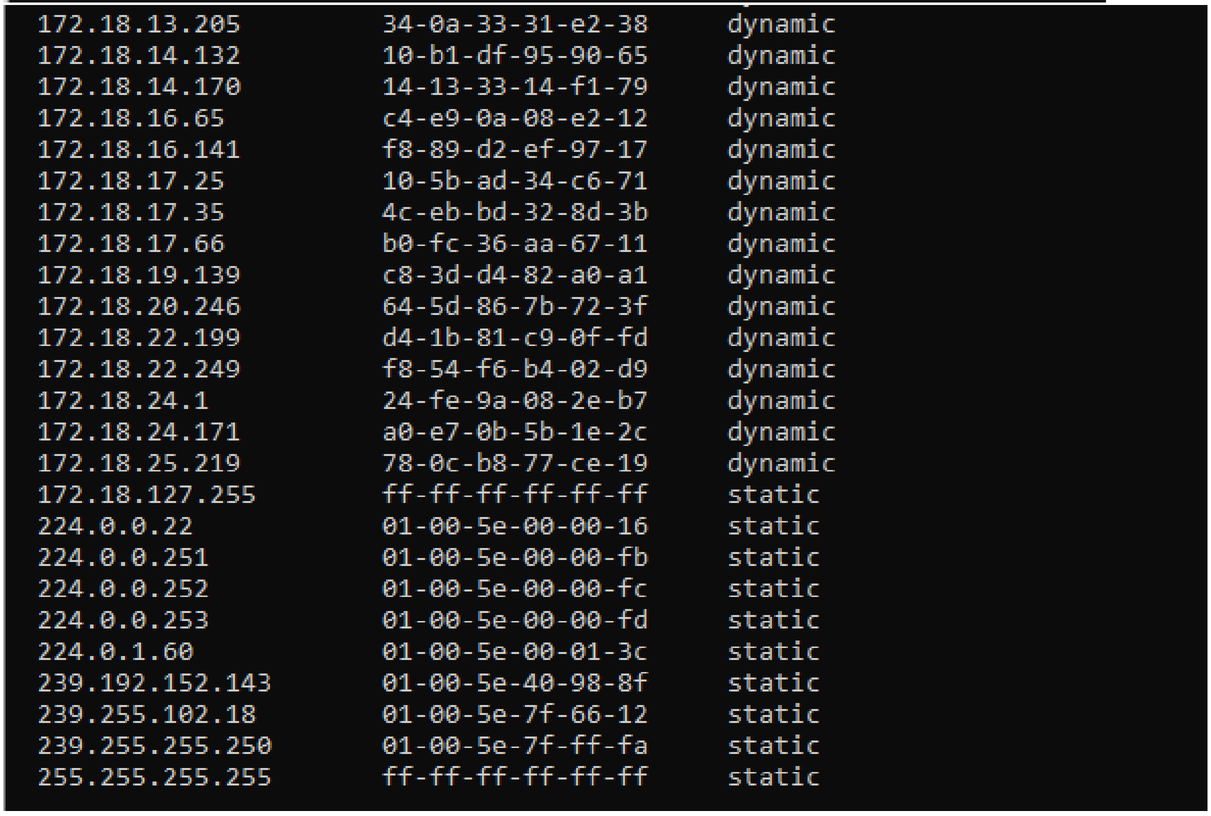
- ➔ Per-hop round-trip time statistics.
- ➔ Packet loss information for each hop.
- ➔ Performance of the entire route.

## ARP

ARP stands for Address Resolution Protocol. This protocol is used to map IP addresses to MAC addresses on a local network. This command is used to view and manage ARP cache on a computer. The ARP cache is a table that keeps track of the mapping between IP addresses and MAC addresses for devices on the local network.



```
C:\Users\Aditi>arp -a

Interface: 192.168.56.1 --- 0x7
  Internet Address      Physical Address      Type
  192.168.56.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  224.0.0.253           01-00-5e-00-00-fd     static
  224.0.1.60            01-00-5e-00-01-3c     static
  239.192.152.143       01-00-5e-40-98-8f     static
  239.255.102.18        01-00-5e-7f-66-12     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 172.18.16.132 --- 0xc
  Internet Address      Physical Address      Type
  169.254.54.55         ac-50-de-b0-d2-28     dynamic
  169.254.86.190        84-69-93-8e-df-cf     dynamic
  169.254.102.195       a4-d7-3c-18-3c-90     dynamic
  169.254.106.136       38-9d-92-ab-41-c7     dynamic
  169.254.147.191       5c-ea-1d-33-10-e4     dynamic
  169.254.241.231       e0-bb-9e-34-21-a2     dynamic
  172.18.0.1            dc-0b-09-7b-d8-ff     dynamic
  172.18.10.90          90-48-9a-5f-bd-4b     dynamic
  172.18.11.46          44-03-2c-ea-2d-7e     dynamic
  172.18.12.119         ac-d1-b8-c3-42-ab     dynamic
  172.18.13.95          60-14-b3-71-1a-cf     dynamic
```

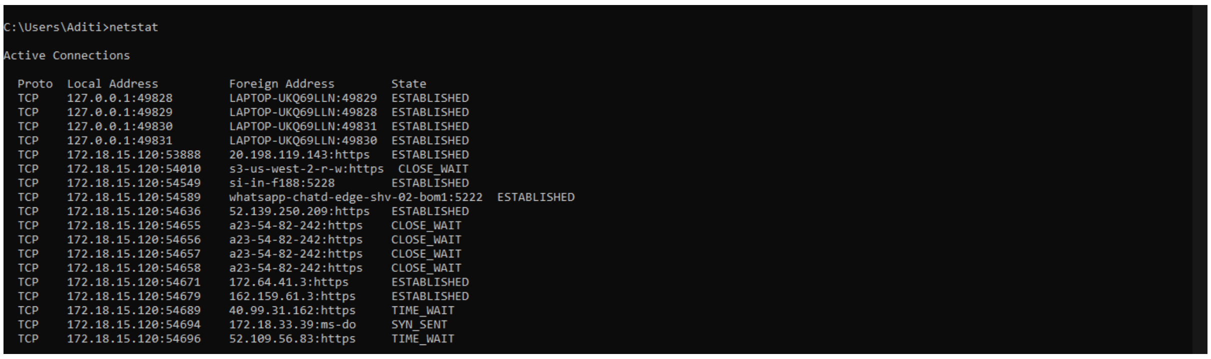


```
  172.18.13.205         34-0a-33-31-e2-38     dynamic
  172.18.14.132         10-b1-df-95-90-65     dynamic
  172.18.14.170         14-13-33-14-f1-79     dynamic
  172.18.16.65          c4-e9-0a-08-e2-12     dynamic
  172.18.16.141         f8-89-d2-ef-97-17     dynamic
  172.18.17.25          10-5b-ad-34-c6-71     dynamic
  172.18.17.35          4c-eb-bd-32-8d-3b     dynamic
  172.18.17.66          b0-fc-36-aa-67-11     dynamic
  172.18.19.139         c8-3d-d4-82-a0-a1     dynamic
  172.18.20.246         64-5d-86-7b-72-3f     dynamic
  172.18.22.199         d4-1b-81-c9-0f-fd     dynamic
  172.18.22.249         f8-54-f6-b4-02-d9     dynamic
  172.18.24.1           24-fe-9a-08-2e-b7     dynamic
  172.18.24.171         a0-e7-0b-5b-1e-2c     dynamic
  172.18.25.219         78-0c-b8-77-ce-19     dynamic
  172.18.127.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  224.0.0.253           01-00-5e-00-00-fd     static
  224.0.1.60            01-00-5e-00-01-3c     static
  239.192.152.143       01-00-5e-40-98-8f     static
  239.255.102.18        01-00-5e-7f-66-12     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

Using the arp -a command, I found I have two interfaces. Interfaces refer to the Network Interface Cards.

## NETSTAT

It displays information about network connections, routing tables, etc. Its useful in monitoring network activities and diagnosing network issues. It basically displays the protocol statistics and current TCP/IP network connections



```
C:\Users\Aditi>netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:49828        LAPTOP-UKQ69LLN:49829  ESTABLISHED
  TCP    127.0.0.1:49829        LAPTOP-UKQ69LLN:49828  ESTABLISHED
  TCP    127.0.0.1:49830        LAPTOP-UKQ69LLN:49831  ESTABLISHED
  TCP    127.0.0.1:49831        LAPTOP-UKQ69LLN:49830  ESTABLISHED
  TCP    172.18.15.120:53888    20.198.119.143:https   ESTABLISHED
  TCP    172.18.15.120:54010    s3-us-west-2-r-w:https  CLOSE_WAIT
  TCP    172.18.15.120:54549    si-in-f188:5228        ESTABLISHED
  TCP    172.18.15.120:54589    whatsapp-chatd-edge-shv-02-bom1:5222  ESTABLISHED
  TCP    172.18.15.120:54636    52.139.250.209:https   ESTABLISHED
  TCP    172.18.15.120:54655    a23-54-82-242:https    CLOSE_WAIT
  TCP    172.18.15.120:54656    a23-54-82-242:https    CLOSE_WAIT
  TCP    172.18.15.120:54657    a23-54-82-242:https    CLOSE_WAIT
  TCP    172.18.15.120:54658    a23-54-82-242:https    CLOSE_WAIT
  TCP    172.18.15.120:54671    172.64.41.3:https      ESTABLISHED
  TCP    172.18.15.120:54679    162.159.61.3:https     ESTABLISHED
  TCP    172.18.15.120:54689    40.99.31.162:https     TIME_WAIT
  TCP    172.18.15.120:54694    172.18.33.39:ms-do     SYN_SENT
  TCP    172.18.15.120:54696    52.109.56.83:https     TIME_WAIT
```

The Proto (Protocol) column shows the network protocol associated with each network connection. It has values like TCP, UDP, ICMP, RAW or IP.

The local address column displays the local endpoint of network connection. It includes the local IP address and port number.

The foreign address column shows the endpoint of network connection which includes remote ip address and port number.

The state column in this case for TCP, indicates the current state of connection i.e.,

LISTEN: The server is listening for incoming connections

ESTABILISHED: A connection is established and data can be exchanged.

TIME_WAIT: The connection is waiting to ensure all data is transmitted.

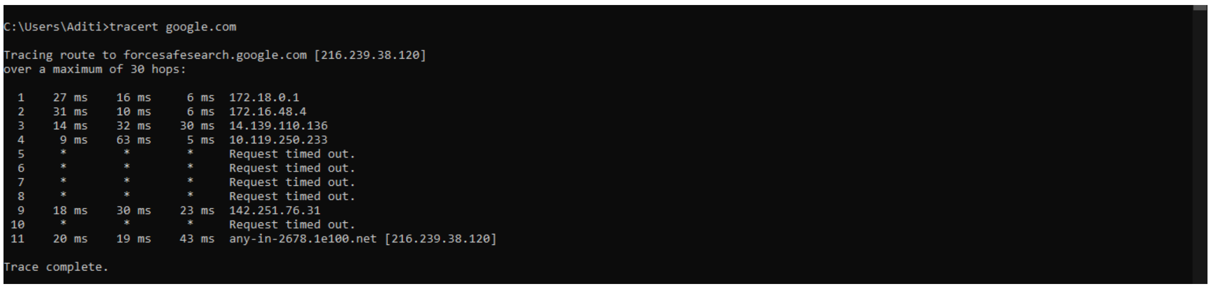
CLOSE_WAIT: The local end of the connection has closed, waiting for the remote end to close.

Options:

- netstat: Displays active network connections
- -a: Shows all connections
- -n: Displays numerical addresses and port numbers
- -r: Displays the routing table
- -s: Displays statistics for various network protocols

## TRACERT

Short for Trace Route, it is a command used to trace the path that data packets take from our computer to a destination host or server. It shows the list of all intermediate routers or hops along with the time it takes for data to read each hop. traceroute is used in linux or mac os. tracert is used in windows.



```
C:\Users\Aditi>tracert google.com

Tracing route to forcesafesearch.google.com [216.239.38.120]
over a maximum of 30 hops:

  1    27 ms    16 ms     6 ms  172.18.0.1
  2    31 ms    10 ms     6 ms  172.16.48.4
  3    14 ms    32 ms    30 ms  14.139.110.136
  4     9 ms    63 ms     5 ms  10.119.250.233
  5     *        *        *     Request timed out.
  6     *        *        *     Request timed out.
  7     *        *        *     Request timed out.
  8     *        *        *     Request timed out.
  9    18 ms    30 ms    23 ms  142.251.76.31
 10     *        *        *     Request timed out.
 11    20 ms    19 ms    43 ms  any-in-2678.1e100.net [216.239.38.120]

Trace complete.
```

The output of this command shows

Hop number: It indicates the order of hop along the route starting from 1 for initial hop (local machine) and increasing with each hop further.

Minimum RTT: It displays in ms the fastest response time observed during the tracert process for that hop.

Maximum RTT: It displays in ms the slowest response time observed during the tracert process for that hop.

Average RTT: It displays the typical response time observed during the tracert process for that hop

IP Address or Hostname: The IP address or hostname of router or device at each hop along the route is displayed.

Request Timed Out: It means that the router or device at a specific hop along the route didn’t respond to tracert request within set timeout period.

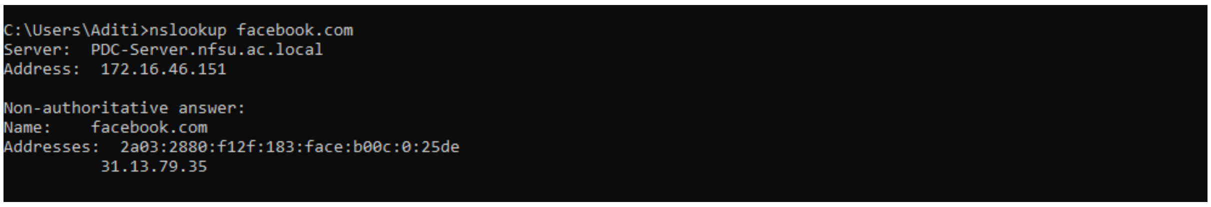
Asterisk (*): Signifies the same condition as request timed out.

Some Options:

- -h [max_hops]: Sets maximum hops to trace
- -w [timeout]: Sets maximum time to wait for each reply

## NSLOOKUP

Short for Name Server Lookup is a command for querying Domain Name System (DNS) servers to obtain information about domain names, IP addresses and DNS records. Nslookup is the name of a command that lets users enter a host name and find out the corresponding IP address or domain name system (DNS) record. Users can also enter a command in nslookup to do a reverse DNS lookup and find the host name for a specified IP address.



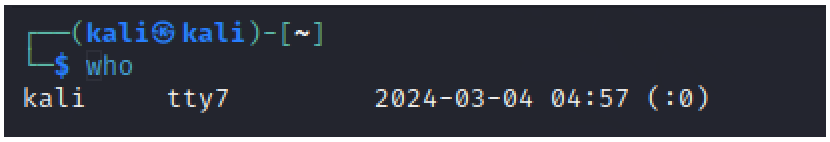
```
C:\Users\Aditi>nslookup facebook.com
Server:  PDC-Server.nfsu.ac.local
Address:  172.16.46.151

Non-authoritative answer:
Name:    facebook.com
Addresses:  2a03:2880:f12f:183:face:b00c:0:25de
          31.13.79.35
```

Some options:

- server [DNS server]: Specifies the DNS server to use for queries
- ls [domain]: Lists information about the DNS domain (zone transfer)

## WHO

The who command is a basic Unix and Linux command that displays information about users who are currently logged into the system. It provides details such as the username, terminal, login time, and originating IP address.



```
┌──(kali㉿kali)-[~]
└─$ who
kali     tty7         2024-03-04 04:57 (:0)
```

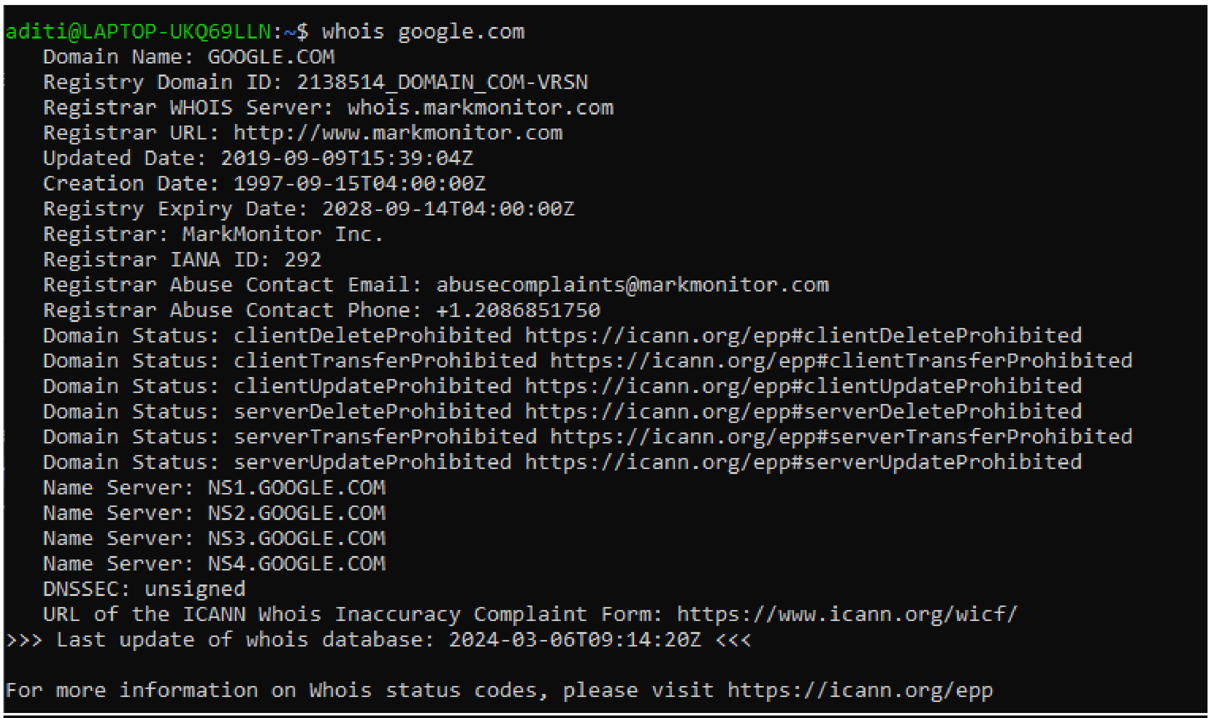
Output is username, Terminal, Login time and session information. Here, the user "kali" is currently logged into the 7th virtual terminal on a graphical environment, and the session started on March 4, 2024, at 04:57 AM.

It is mostly used for:

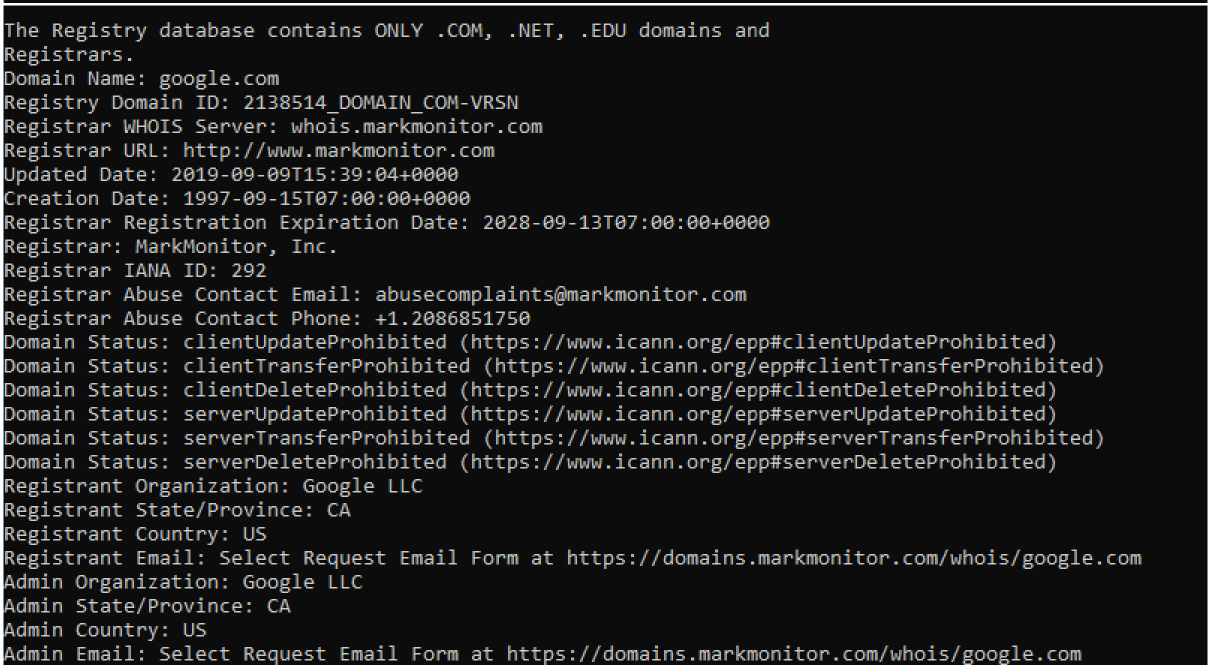
- Checking who is currently using the system.
- Verifying active user sessions.
- Identifying the terminal or device each user is logged into.
- Monitoring login times.

## WHOIS

The whois command is a useful tool for obtaining information about domain names, IP Addresses, and network devices registered with ICANN (Internet Corporation for Assigned Names and Numbers). whois command is a simple and powerful tool that can be useful for network administration, web development, and security tasks.



```
aditi@LAPTOP-UKQ69LLN:~$ whois google.com
   Domain Name: GOOGLE.COM
   Registry Domain ID: 2138514_DOMAIN_COM-VRSN
   Registrar WHOIS Server: whois.markmonitor.com
   Registrar URL: http://www.markmonitor.com
   Updated Date: 2019-09-09T15:39:04Z
   Creation Date: 1997-09-15T04:00:00Z
   Registry Expiry Date: 2028-09-14T04:00:00Z
   Registrar: MarkMonitor Inc.
   Registrar IANA ID: 292
   Registrar Abuse Contact Email: abusecomplaints@markmonitor.com
   Registrar Abuse Contact Phone: +1.2086851750
   Domain Status: clientDeleteProhibited https://icann.org/epp#clientDeleteProhibited
   Domain Status: clientTransferProhibited https://icann.org/epp#clientTransferProhibited
   Domain Status: clientUpdateProhibited https://icann.org/epp#clientUpdateProhibited
   Domain Status: serverDeleteProhibited https://icann.org/epp#serverDeleteProhibited
   Domain Status: serverTransferProhibited https://icann.org/epp#serverTransferProhibited
   Domain Status: serverUpdateProhibited https://icann.org/epp#serverUpdateProhibited
   Name Server: NS1.GOOGLE.COM
   Name Server: NS2.GOOGLE.COM
   Name Server: NS3.GOOGLE.COM
   Name Server: NS4.GOOGLE.COM
   DNSSEC: unsigned
   URL of the ICANN Whois Inaccuracy Complaint Form: https://www.icann.org/wicf/
>>> Last update of whois database: 2024-03-06T09:14:20Z <<<

For more information on Whois status codes, please visit https://icann.org/epp
```



```
The Registry database contains ONLY .COM, .NET, .EDU domains and
Registrars.
Domain Name: google.com
Registry Domain ID: 2138514_DOMAIN_COM-VRSN
Registrar WHOIS Server: whois.markmonitor.com
Registrar URL: http://www.markmonitor.com
Updated Date: 2019-09-09T15:39:04+0000
Creation Date: 1997-09-15T07:00:00+0000
Registrar Registration Expiration Date: 2028-09-13T07:00:00+0000
Registrar: MarkMonitor, Inc.
Registrar IANA ID: 292
Registrar Abuse Contact Email: abusecomplaints@markmonitor.com
Registrar Abuse Contact Phone: +1.2086851750
Domain Status: clientUpdateProhibited (https://www.icann.org/epp#clientUpdateProhibited)
Domain Status: clientTransferProhibited (https://www.icann.org/epp#clientTransferProhibited)
Domain Status: clientDeleteProhibited (https://www.icann.org/epp#clientDeleteProhibited)
Domain Status: serverUpdateProhibited (https://www.icann.org/epp#serverUpdateProhibited)
Domain Status: serverTransferProhibited (https://www.icann.org/epp#serverTransferProhibited)
Domain Status: serverDeleteProhibited (https://www.icann.org/epp#serverDeleteProhibited)
Registrant Organization: Google LLC
Registrant State/Province: CA
Registrant Country: US
Registrant Email: Select Request Email Form at https://domains.markmonitor.com/whois/google.com
Admin Organization: Google LLC
Admin State/Province: CA
Admin Country: US
Admin Email: Select Request Email Form at https://domains.markmonitor.com/whois/google.com
```

The output includes details such as the registrar, registration status, name servers, and contact information for the domain owner and administrators.

## NCAT

It is a command line tool also known as netcat's advanced version which can be used to estabilish network connections as client/server and perform simple data transfer using various protocols. We can also transfer files using ncat between systems both plain text and binary files.

Here I have given an example of transferring a file using netcat between two VMs.

Commands used here are:

1) nc -lvp <port-number> > file_name

This command makes the machine act as server to listen the requests and respond. By the command

"> file_name" it adds the data sent from the client to the file

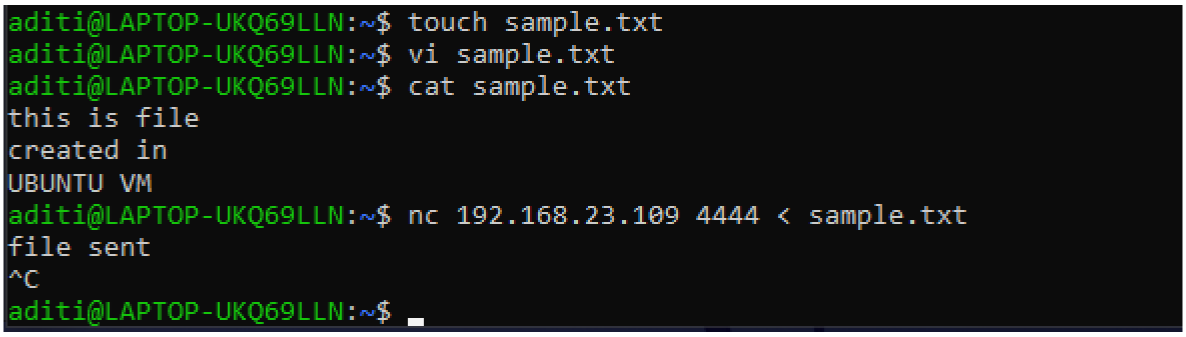
2) nc <ip-address> <port-number> < file_name

This command makes machine act as a client to send requests to the server. By the command

"< file_name" it sends the file to the server. And Ip address is of the listening machine

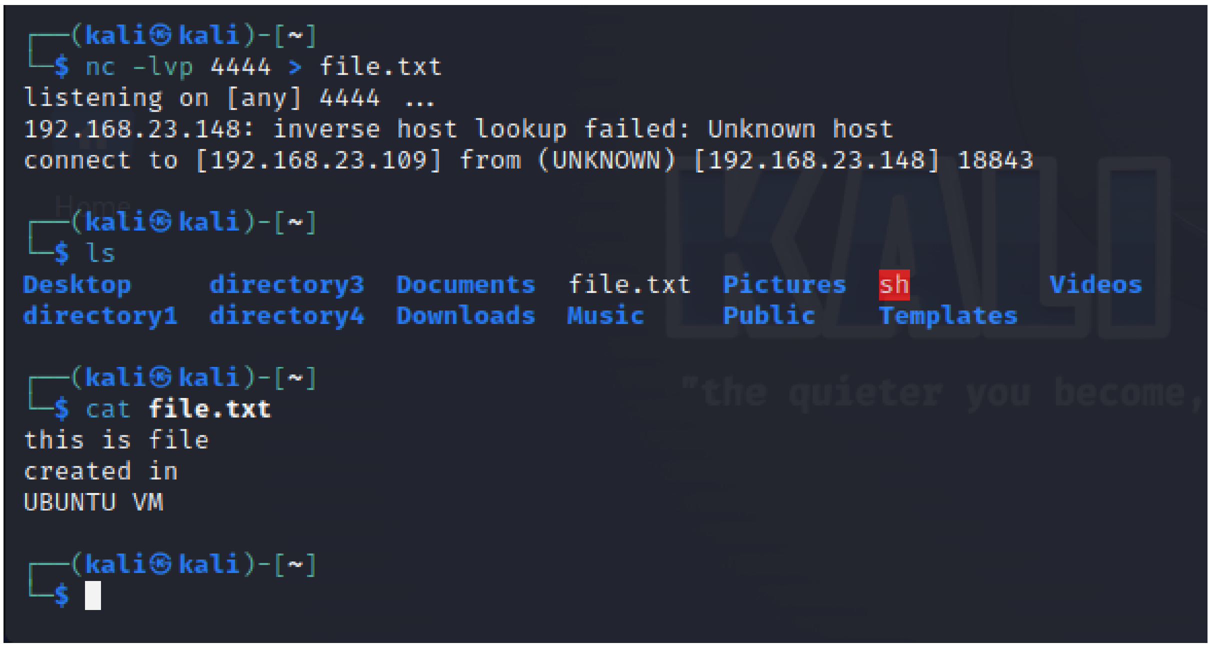
Here the file created in ubuntu sample.txt is transferred to kali linux and stored as file.txt through netcat. We can do the vice-versa as well.

Ubuntu VM (Client-Sending) [*ip-address = 192.168.23.148*]



```
aditi@LAPTOP-UKQ69LLN:~$ touch sample.txt
aditi@LAPTOP-UKQ69LLN:~$ vi sample.txt
aditi@LAPTOP-UKQ69LLN:~$ cat sample.txt
this is file
created in
UBUNTU VM
aditi@LAPTOP-UKQ69LLN:~$ nc 192.168.23.109 4444 < sample.txt
file sent
^C
aditi@LAPTOP-UKQ69LLN:~$ 
```

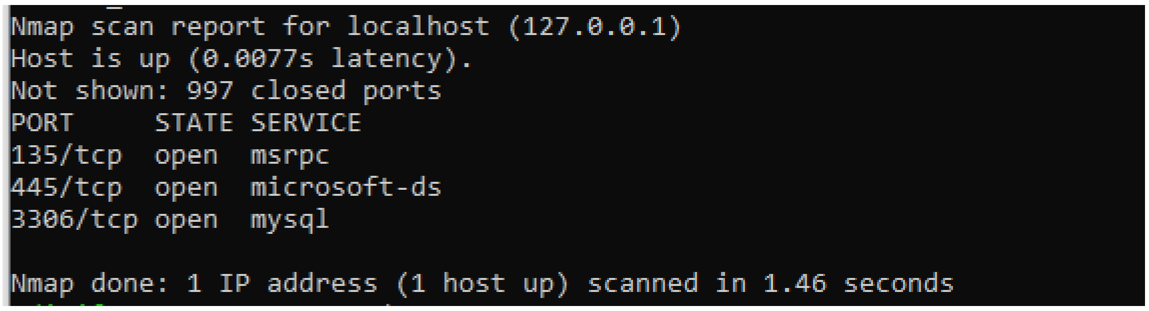
Kali Linux VM (Server-Listening) [*ip-address = 192.168.23.109*]



```
┌──(kali㉿kali)-[~]
└─$ nc -lvp 4444 > file.txt
listening on [any] 4444 ...
192.168.23.148: inverse host lookup failed: Unknown host
connect to [192.168.23.109] from (UNKNOWN) [192.168.23.148] 18843

┌──(kali㉿kali)-[~]
└─$ ls
Desktop      directory3  Documents  file.txt  Pictures  sh         Videos
directory1   directory4  Downloads  Music     Public    Templates

┌──(kali㉿kali)-[~]
└─$ cat file.txt
this is file
created in
UBUNTU VM

┌──(kali㉿kali)-[~]
└─$ 
```

## NMAP

It is a free and open-source tool for vulnerability scanning and network discovery. It can used to monitor single host as well as vast networks with 100s of 1000s of devices and subnets. It scans network and outputs list of ports open/closed, device names, operating systems of each device, services on the ports, and other details.



```
Nmap scan report for localhost (127.0.0.1)
Host is up (0.0077s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
135/tcp  open  msrpc
445/tcp  open  microsoft-ds
3306/tcp open  mysql

Nmap done: 1 IP address (1 host up) scanned in 1.46 seconds
```

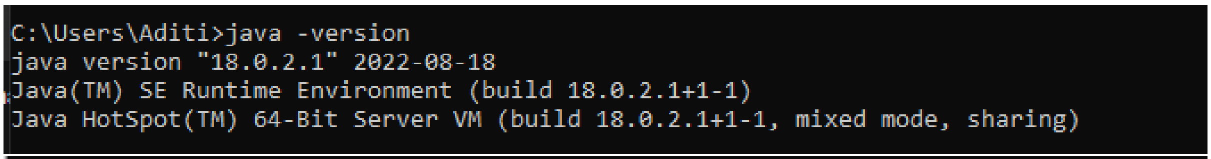
Here a single target scan is performed using ubuntu wsl. It is a basic scan which generally tells online hosts and list of open ports with their services.

Can scan

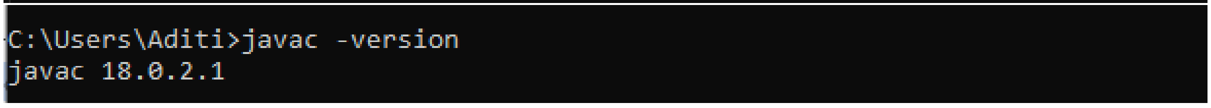
- ➔ single target: nmap [target]
- ➔ multiple target: nmap [target1] [target2]
- ➔ range of ip address to scan: nmap [range of IP]
- ➔ scan entire subnet: nmap [Network/CIDR]
- ➔ scan list of targets: nmap -iL list.txt

## L.2 Install java JDK and JRE both on Ubuntu and windows both and configure the path both in both system and try to understand the all executable programs and understand java commands. Install tomcat and understand server side programming.

Java on Windows:



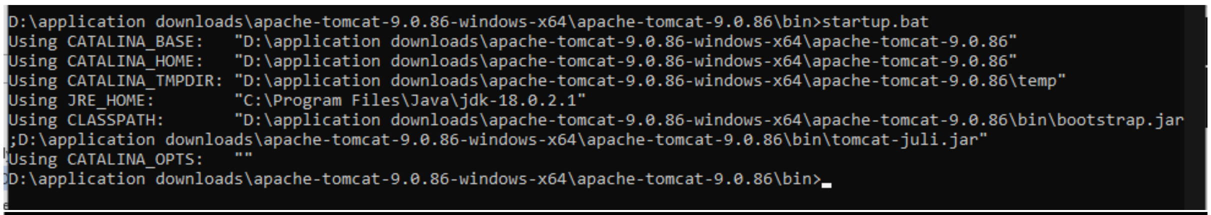
```
C:\Users\Aditi>java -version
java version "18.0.2.1" 2022-08-18
Java(TM) SE Runtime Environment (build 18.0.2.1+1-1)
Java HotSpot(TM) 64-Bit Server VM (build 18.0.2.1+1-1, mixed mode, sharing)
```



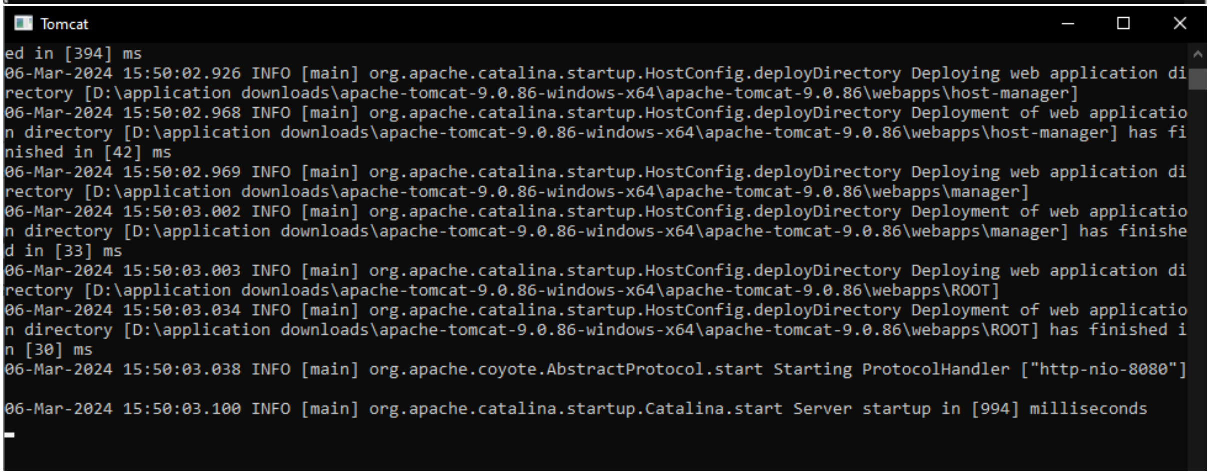
```
C:\Users\Aditi>javac -version
javac 18.0.2.1
```

Installing Tomcat

1. Downloaded the zip file of Apache Tomcat from https://tomcat.apache.org/download-90.cgi
2. Extracted the zip file and using cmd executed the “startup.bat” file in the bin directory.



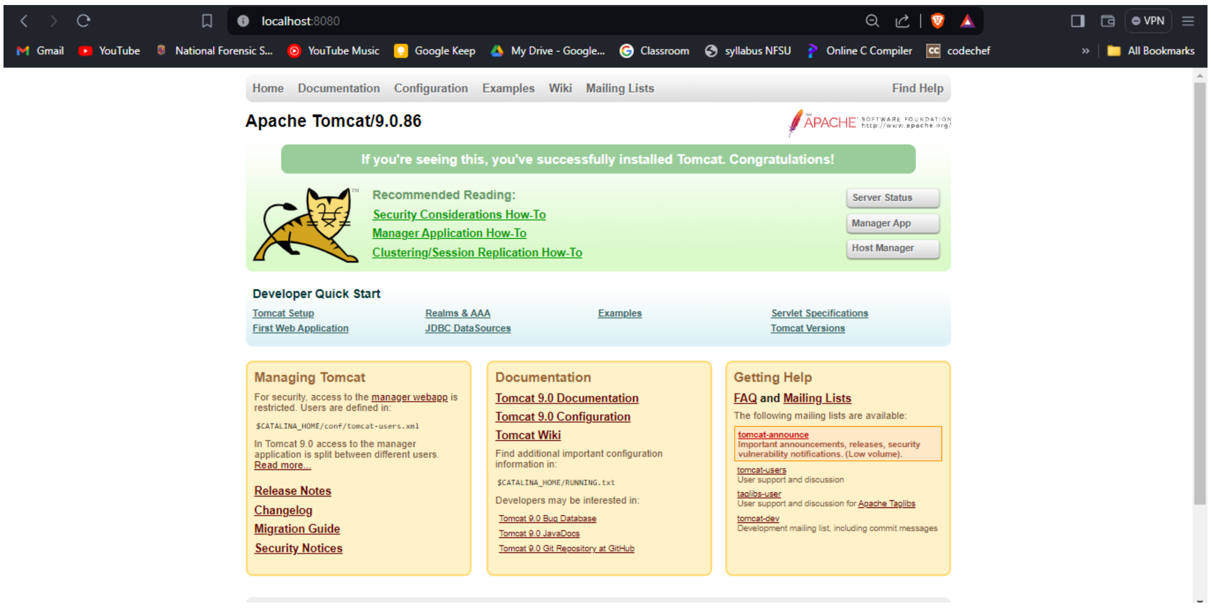
```
D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\bin>startup.bat
Using CATALINA_BASE:   "D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86"
Using CATALINA_HOME:   "D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86"
Using CATALINA_TMPDIR: "D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\temp"
Using JRE_HOME:        "C:\Program Files\Java\jdk-18.0.2.1"
Using CLASSPATH:       "D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\bin\bootstrap.jar
;D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\bin\tomcat-juli.jar"
Using CATALINA_OPTS:   ""
D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\bin>
```



Tomcat

```
ed in [394] ms
06-Mar-2024 15:50:02.926 INFO [main] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application di
rectory [D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\webapps\host-manager]
06-Mar-2024 15:50:02.968 INFO [main] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web applicatio
n directory [D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\webapps\host-manager] has fi
nished in [42] ms
06-Mar-2024 15:50:02.969 INFO [main] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application di
rectory [D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\webapps\manager]
06-Mar-2024 15:50:03.002 INFO [main] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web applicatio
n directory [D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\webapps\manager] has finishe
d in [33] ms
06-Mar-2024 15:50:03.003 INFO [main] org.apache.catalina.startup.HostConfig.deployDirectory Deploying web application di
rectory [D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\webapps\ROOT]
06-Mar-2024 15:50:03.034 INFO [main] org.apache.catalina.startup.HostConfig.deployDirectory Deployment of web applicatio
n directory [D:\application downloads\apache-tomcat-9.0.86-windows-x64\apache-tomcat-9.0.86\webapps\ROOT] has finished i
n [30] ms
06-Mar-2024 15:50:03.038 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-nio-8080"]

06-Mar-2024 15:50:03.100 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in [994] milliseconds
```

3. Accessed the Tomcat server on “localhost:8080” using a web browser.



Apache Tomcat/9.0.86

If you're seeing this, you've successfully installed Tomcat. Congratulations!

Recommended Reading:
Security Considerations How-To
Manager Application How-To
Clustering/Session Replication How-To

Developer Quick Start

|  |  |  |  |
|---|---|---|---|
| Tomcat Setup | Realms & AAA | Examples | Servlet Specifications |
| First Web Application | JDBC DataSources |  | Tomcat Versions |

Managing Tomcat

For security, access to the manager webapp is restricted. Users are defined in:

```
$CATALINA_HOME/conf/tomcat-users.xml
```

In Tomcat 9.0 access to the manager application is split between different users. Read more...

Release Notes
Changelog
Migration Guide
Security Notices

Documentation

Tomcat 9.0 Documentation
Tomcat 9.0 Configuration
Tomcat Wiki

Find additional important configuration information in:

```
$CATALINA_HOME/RUNNING.txt
```

Developers may be interested in:

Tomcat 9.0 Bug Database
Tomcat 9.0 JavaDocs
Tomcat 9.0 Git Repository at GitHub

Getting Help

FAQ and Mailing Lists

The following mailing lists are available:

tomcat-announce
Important announcements, releases, security vulnerability notifications. (Low volume).

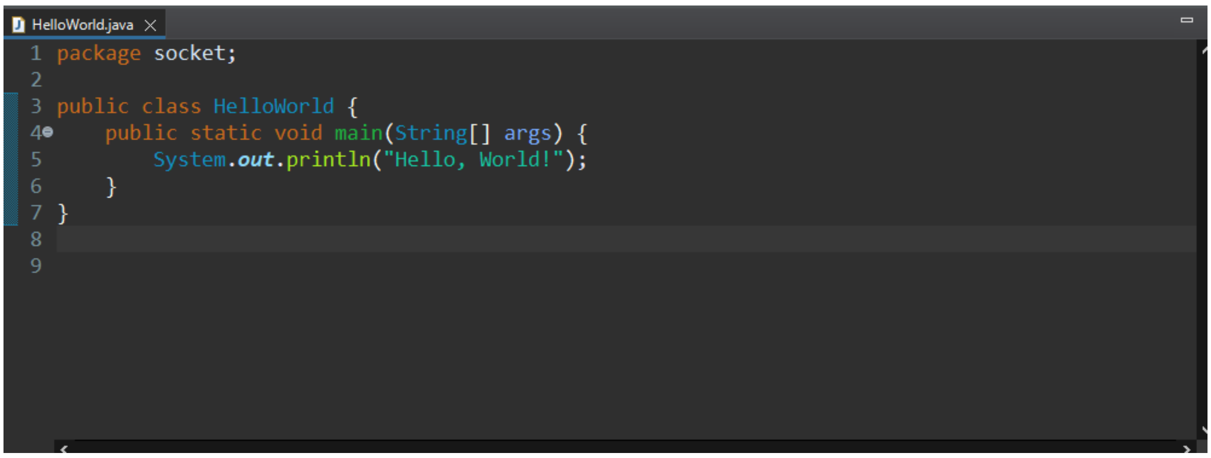
tomcat-users
User support and discussion

taglibs-user
User support and discussion for Apache Taglibs

tomcat-dev
Development mailing list, including commit messages

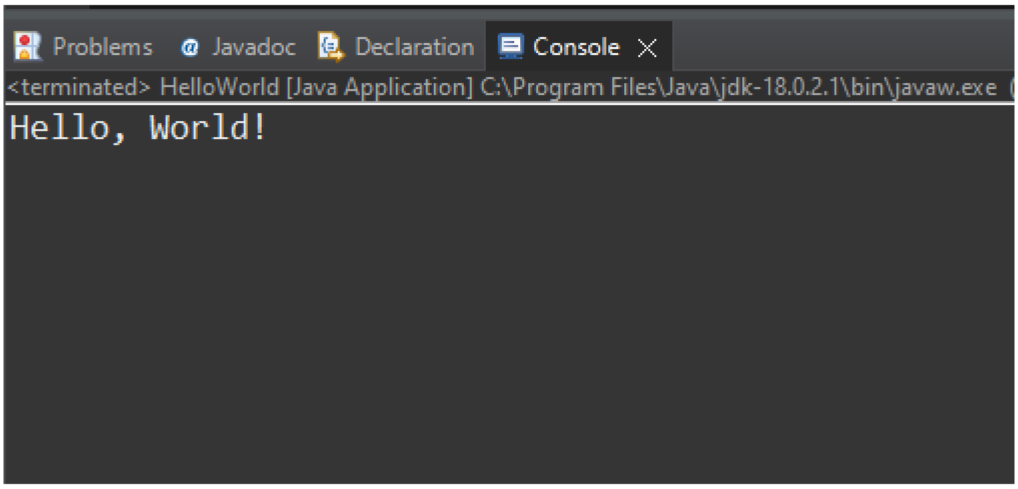
## L.3 Install Eclipse or Netbeans and configure basic java and J2EE tomcat server configure and develop basic java and HTML program in either Eclipse or Netbeans IDE

Code:



```java
package socket;

public class HelloWorld {
    public static void main(String[] args) {
        System.out.println("Hello, World!");
    }
}
```

Output:



```
<terminated> HelloWorld [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (
Hello, World!
```

## L.4 Develop Java based Login application and provide hard coded login name and password along with remember facility and show successful login and failure login both cases

Code:

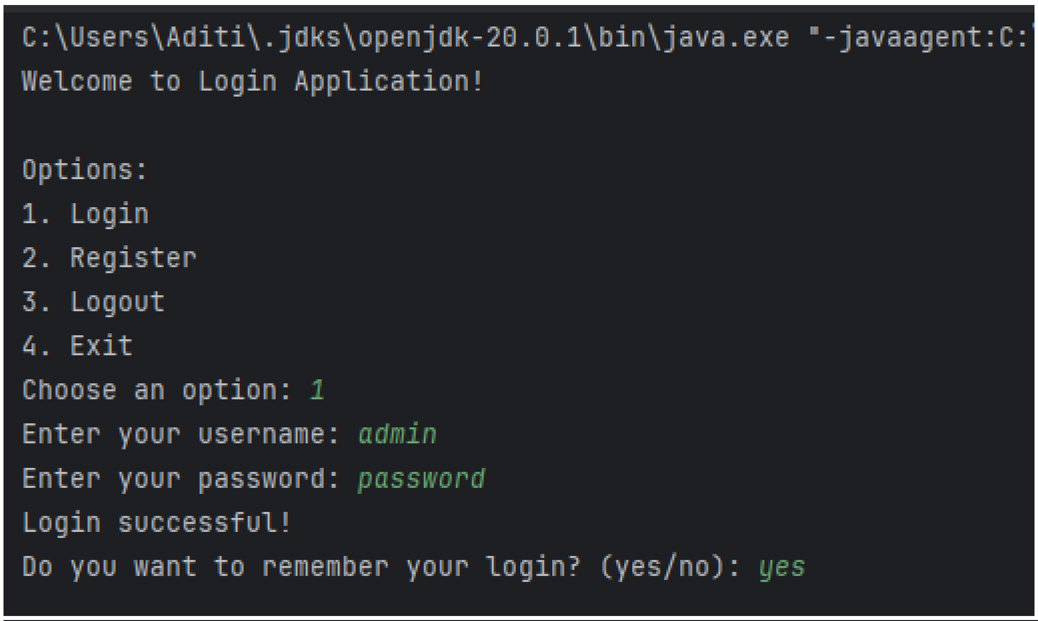
```
import java.util.ArrayList;

import java.util.Scanner;


public class LoginApplication {

   // login credentials list

   static ArrayList<String> usernames = new ArrayList<String>();

   static ArrayList<String> passwords = new ArrayList<String>();




   private static boolean rememberMe = false;

   private static String loggedInUsername = "";


   public static void main(String[] args) {

      //Hardcoded username and password

      usernames.add("admin");

      passwords.add("password");

      Scanner scanner = new Scanner(System.in);


      System.out.println("Welcome to Login Application!");


      while (true) {

         System.out.println("\nOptions:");

         System.out.println("1. Login");

         System.out.println("2. Register");

         System.out.println("3. Logout");

         System.out.println("4. Exit");

         System.out.print("Choose an option: ");
```

```
            int choice = scanner.nextInt();
            scanner.nextLine(); // Consume newline

            switch (choice) {
                case 1:
                    login(scanner);
                    break;
                case 2:
                    register(scanner);
                    break;
                case 3:
                    logout();
                    break;
                case 4:
                    System.out.println("Exiting...");
                    scanner.close();
                    System.exit(0);
                default:
                    System.out.println("Invalid option. Please try again.");
            }
        }
    }

    // Method to handle login
    private static void login(Scanner scanner) {
        if (!loggedInUsername.isEmpty()) {
            System.out.println("You are already logged in as " + loggedInUsername);
            return;
        }

        System.out.print("Enter your username: ");
        String enteredUsername = scanner.nextLine();
        System.out.print("Enter your password: ");
        String enteredPassword = scanner.nextLine();

        // Validate credentials
        if (validateCredentials(enteredUsername, enteredPassword)) {
            loggedInUsername = enteredUsername;
            System.out.println("Login successful!");

            // Remember login if requested
            System.out.print("Do you want to remember your login? (yes/no): ");
            String rememberChoice = scanner.nextLine();
            if (rememberChoice.equalsIgnoreCase("yes")) {
                rememberMe = true;
            } else {
                rememberMe = false;
            }
        } else {
            System.out.println("Login failed. Incorrect username or password.");
        }
    }

    // Method to handle registration
    private static void register(Scanner scanner) {
        if (!loggedInUsername.isEmpty()) {
            System.out.println("You are already logged in as " + loggedInUsername);
            return;
        }

        System.out.print("Enter a new username: ");
        String newUsername = scanner.nextLine();
        System.out.print("Enter a new password: ");
        String newPassword = scanner.nextLine();

        // store the new username and password in the list
        usernames.add(newUsername);
        passwords.add(newPassword);

        System.out.println("Registration successful! You can now login with your new credentials.");
    }

    // Method to handle logout
    private static void logout() {
        if (!loggedInUsername.isEmpty()) {
            System.out.println("Logging out user: " + loggedInUsername);
            if (!rememberMe) {
                loggedInUsername = "";
            }
        } else {
            System.out.println("You are not currently logged in.");
        }
    }

    // Method to validate entered credentials
    private static boolean validateCredentials(String username, String password) {
        for (int i = 0; i < usernames.size(); i++) {
            if (username.equals(usernames.get(i)) && password.equals(passwords.get(i))) {
                return true;
            }
        }
        return false;
    }
```

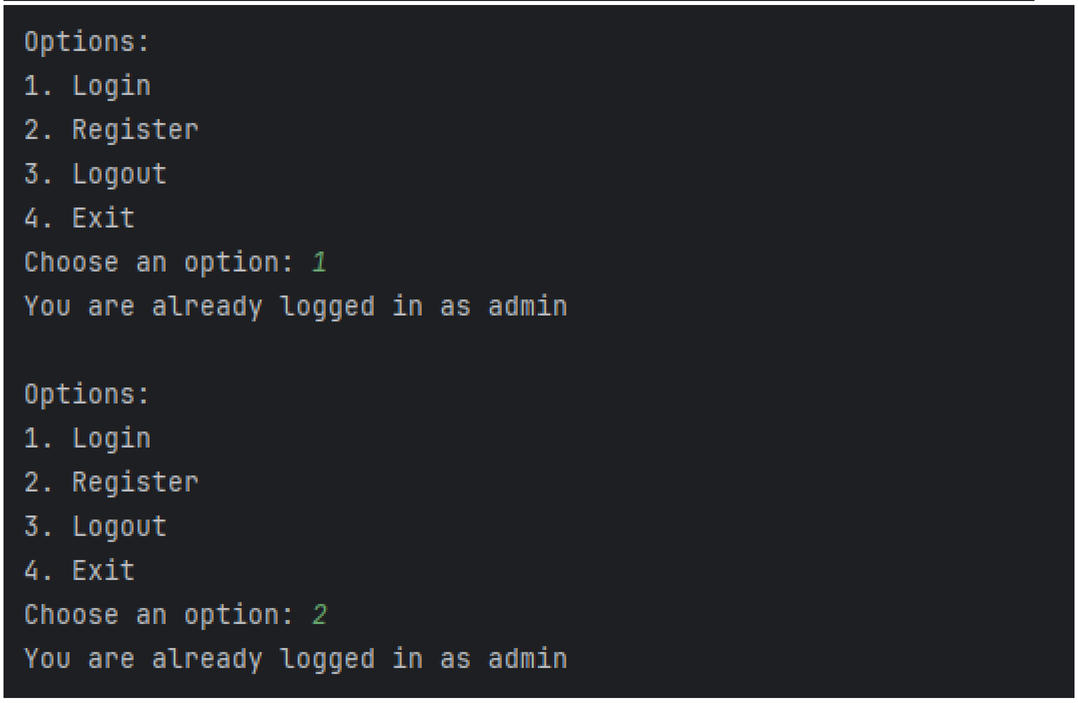
}

Output:

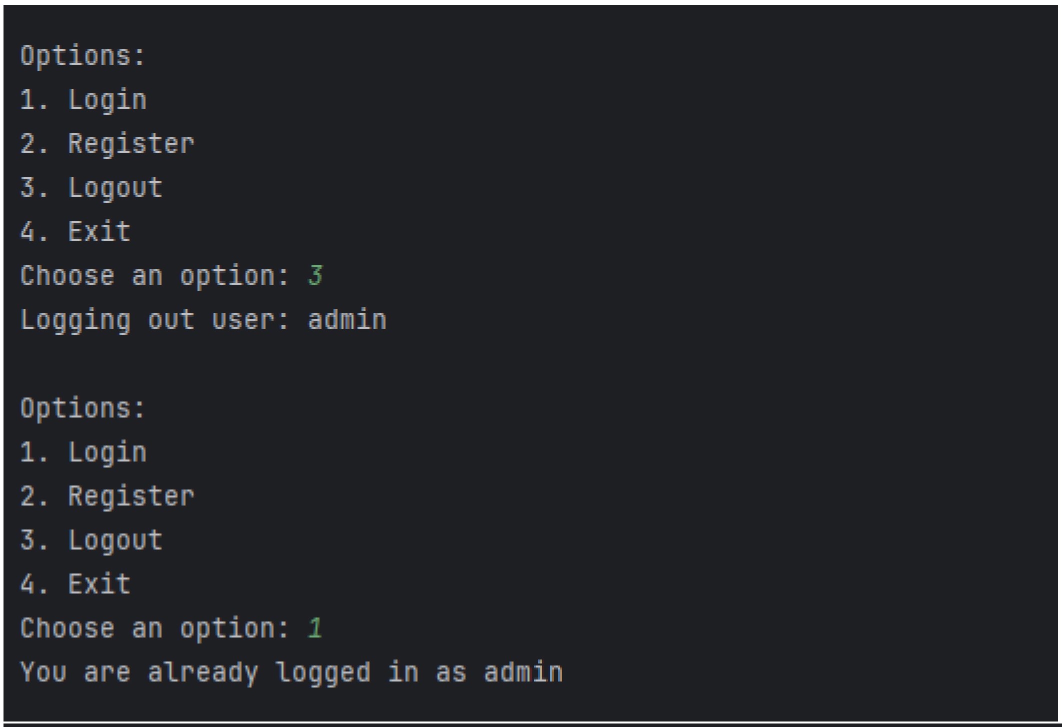
Successful Login and Remember Me enabled.



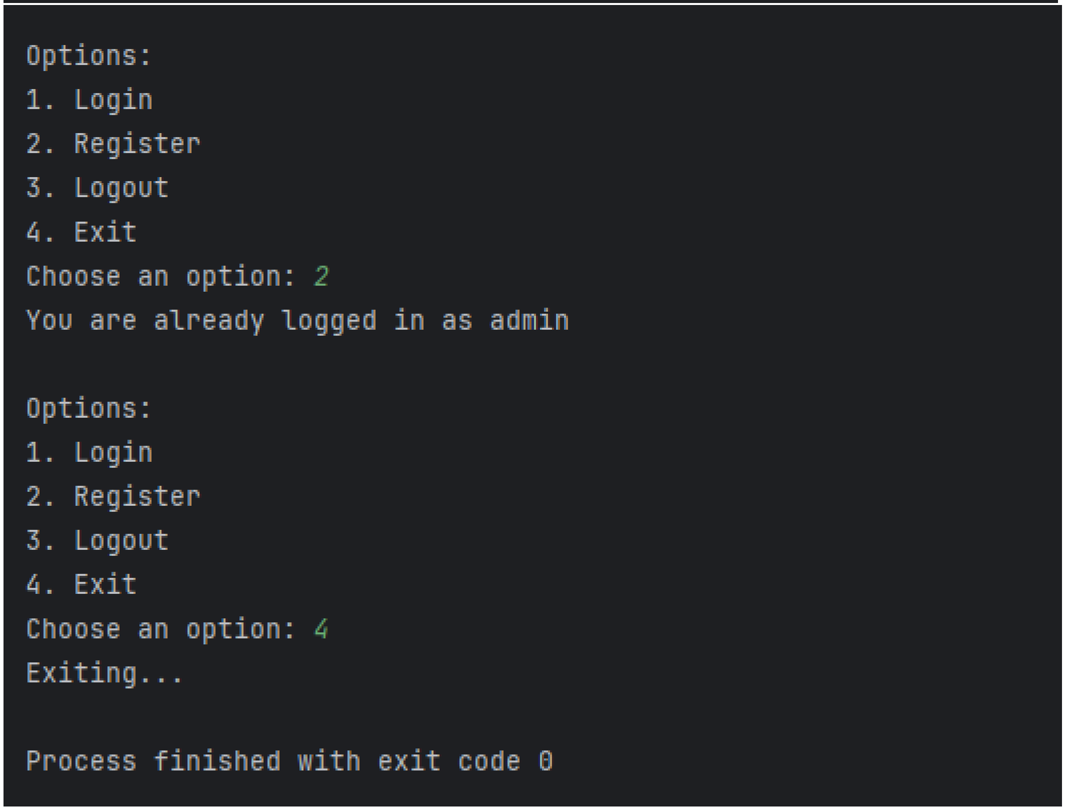
```
C:\Users\Aditi\.jdks\openjdk-20.0.1\bin\java.exe "-javaagent:C:
Welcome to Login Application!

Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 1
Enter your username: admin
Enter your password: password
Login successful!
Do you want to remember your login? (yes/no): yes
```



```
Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 1
You are already logged in as admin

Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 2
You are already logged in as admin
```

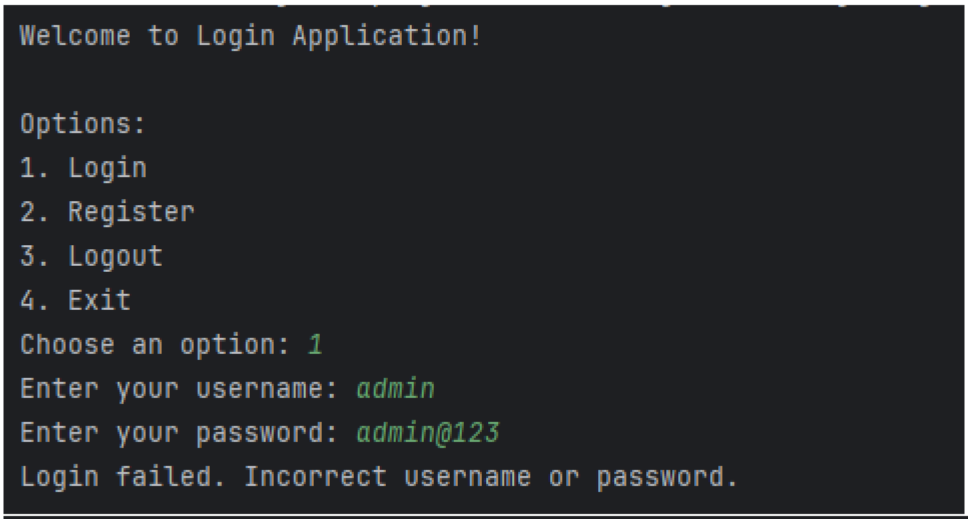


```
Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 3
Logging out user: admin

Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 1
You are already logged in as admin
```

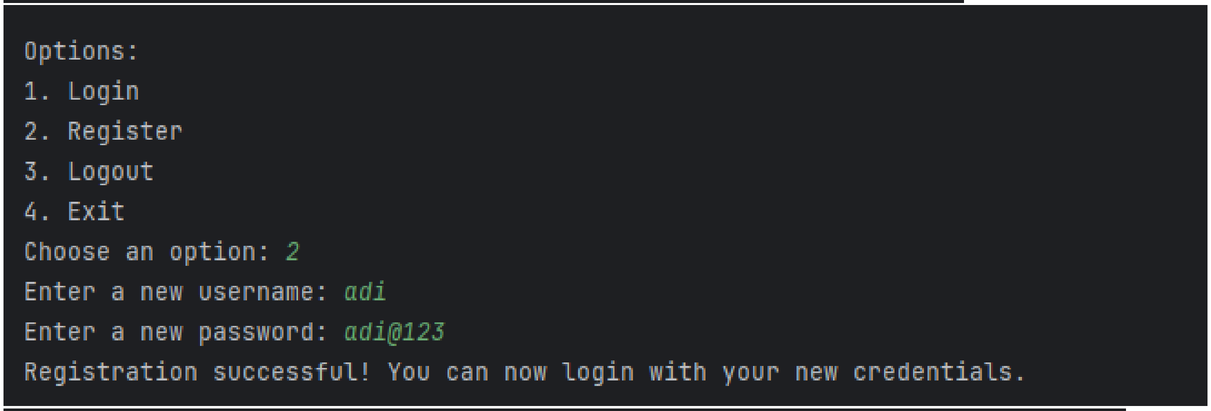


```
Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 2
You are already logged in as admin

Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 4
Exiting...

Process finished with exit code 0
```

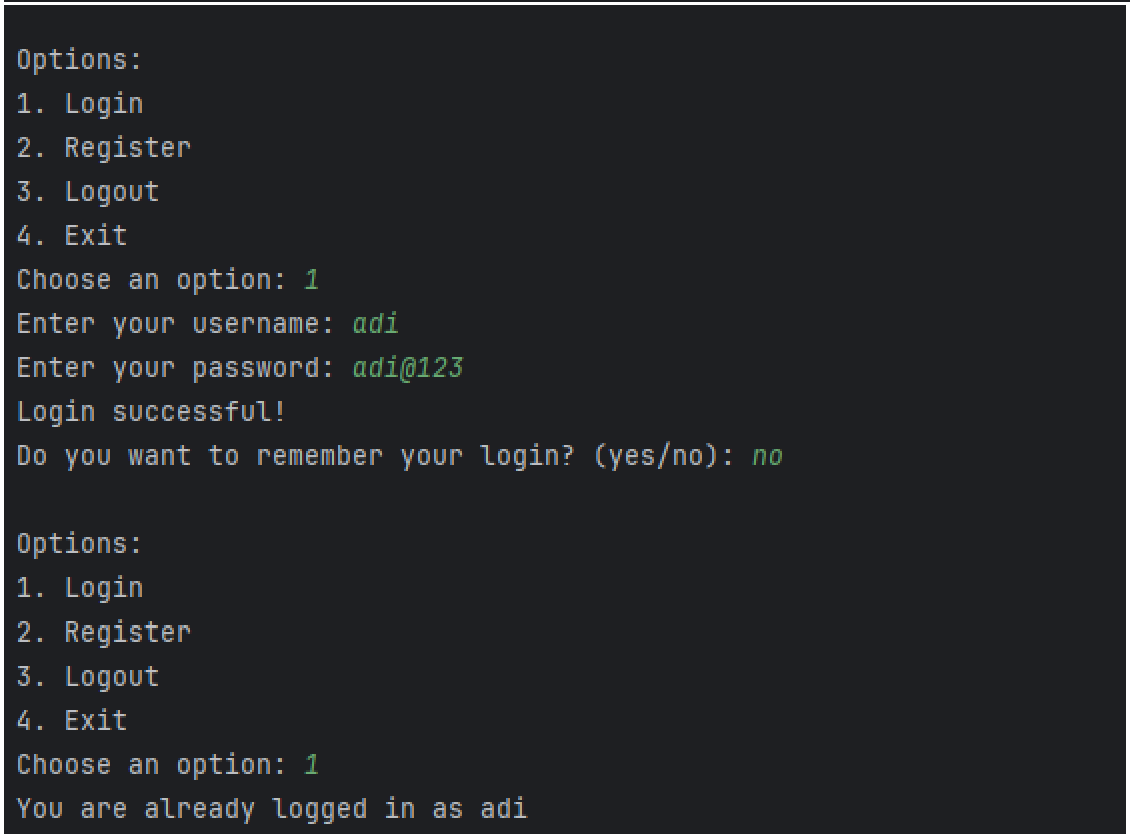
Login Failure and Remember Me disabled.



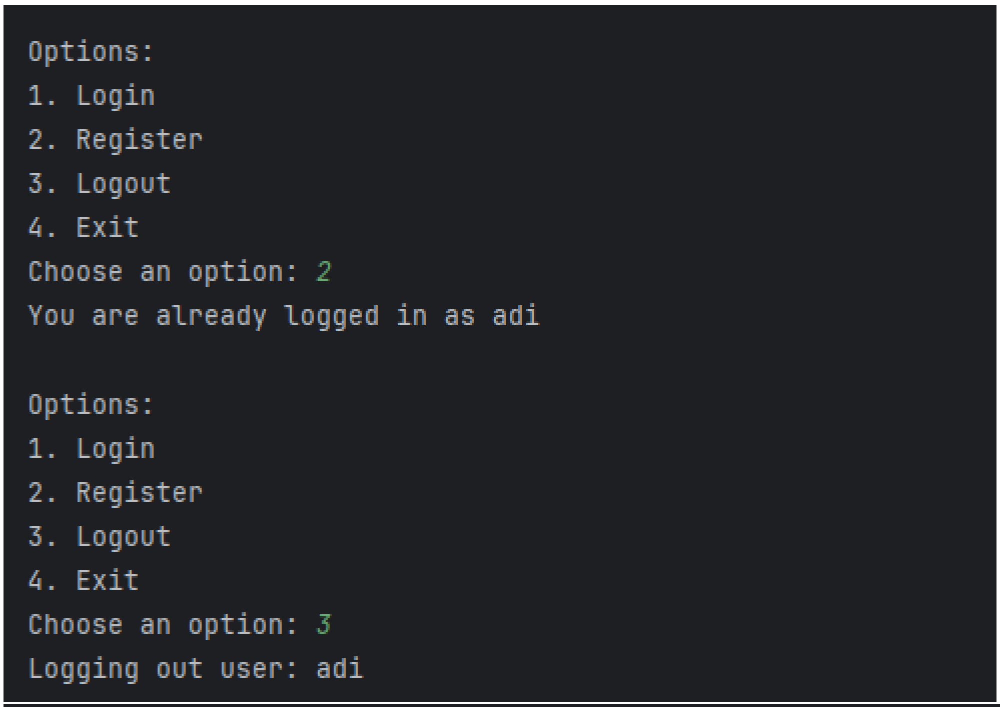
```
Welcome to Login Application!

Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 1
Enter your username: admin
Enter your password: admin@123
Login failed. Incorrect username or password.
```



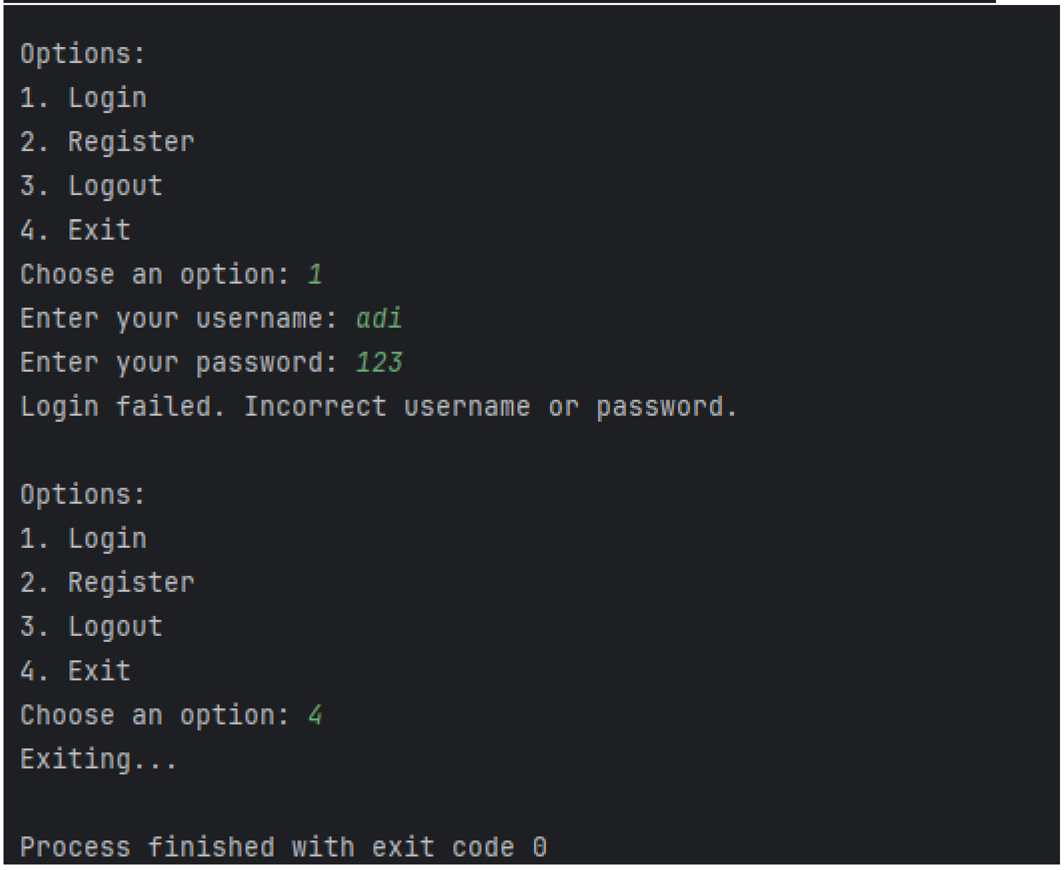
```
Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 2
Enter a new username: adi
Enter a new password: adi@123
Registration successful! You can now login with your new credentials.
```



```
Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 1
Enter your username: adi
Enter your password: adi@123
Login successful!
Do you want to remember your login? (yes/no): no

Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 1
You are already logged in as adi
```



```
Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 2
You are already logged in as adi

Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 3
Logging out user: adi
```



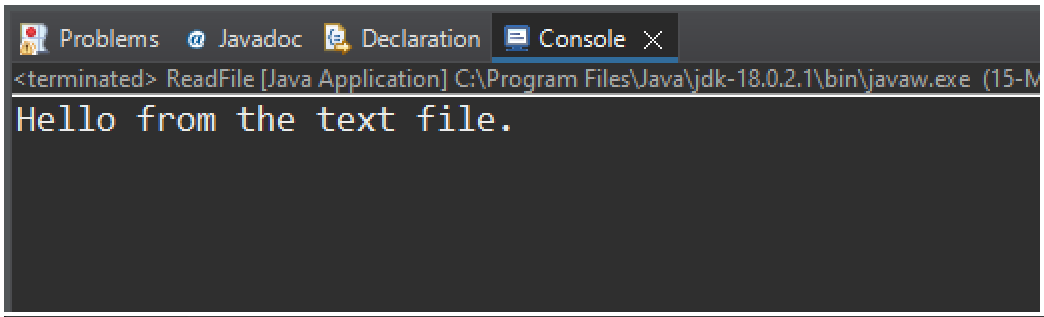
```
Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 1
Enter your username: adi
Enter your password: 123
Login failed. Incorrect username or password.

Options:
1. Login
2. Register
3. Logout
4. Exit
Choose an option: 4
Exiting...

Process finished with exit code 0
```

## L.5 Develop Java programs using Java.io classes for displaying the contents of text file and also write java Program for copying the file facility.

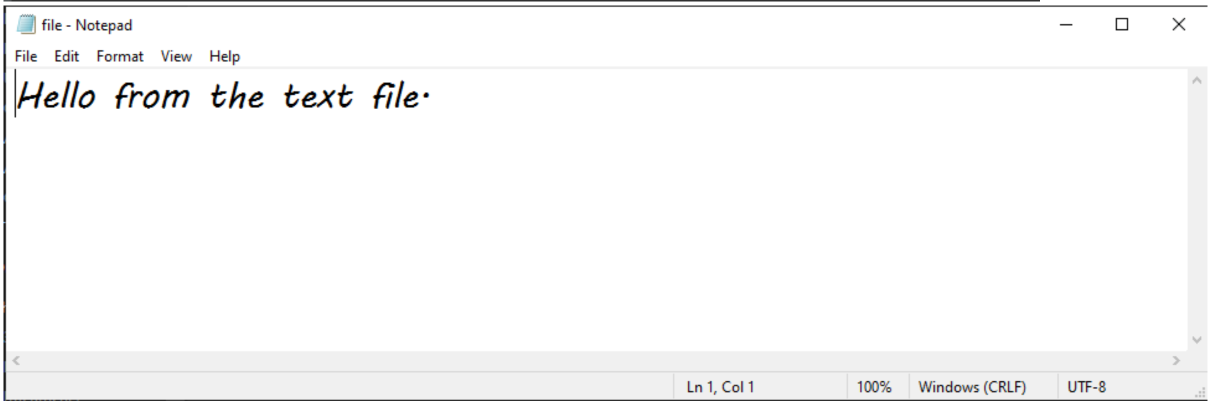
Code:

```
package socket;

import java.io.File;  // Import the File class

import java.io.FileNotFoundException;  // Import this class to handle errors

import java.util.Scanner; // Import the Scanner class to read text files


public class ReadFile {

  public static void main(String[] args) {

    try {

      File myObj = new File(System.getProperty("user.dir") + "/file.txt");

      Scanner myReader = new Scanner(myObj);

      while (myReader.hasNextLine()) {

        String data = myReader.nextLine();

        System.out.println(data);

      }

      myReader.close();

    } catch (FileNotFoundException e) {

      System.out.println("An error occurred.");

      e.printStackTrace();

    }

  }

}
```

Output:



```
<terminated> ReadFile [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe  (15-M
Hello from the text file.
```



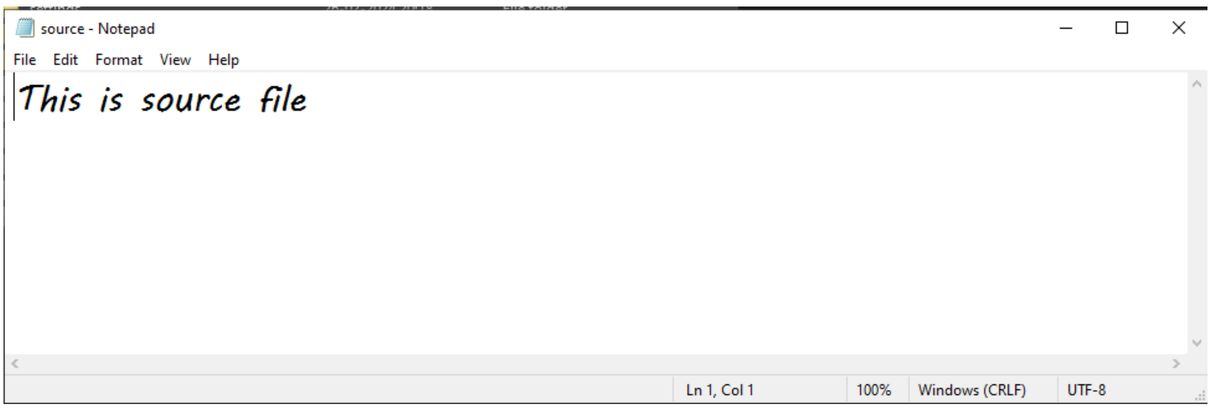
```
Hello from the text file.
```

Code:

```
// Java program to copy content from

// one file to another


import java.io.*;

import java.util.*;


public class CopyFile {


        public static void copyContent(File a, File b)

                throws Exception

        {

                FileInputStream in = new FileInputStream(a);

                FileOutputStream out = new FileOutputStream(b);


                try {


                        int n;
                        // read() function to read the
                        // byte of data
                        while ((n = in.read()) != -1) {
                                // write() function to write
                                // the byte of data
                                out.write(n);
                        }
                }
                finally {
                        if (in != null) {

                                // close() function to close the
                                // stream
                                in.close();
                        }
                        // close() function to close
                        // the stream
                        if (out != null) {
                                out.close();
                        }
                }
                System.out.println("File Copied");
        }

        public static void main(String[] args) throws Exception
        {
                // source file
                File x = new File(System.getProperty("user.dir") + "/source.txt");

                // destination file
```

```
                File y = new File(System.getProperty("user.dir") + "/destination.txt");

                // method called to copy the
                // contents from x to y
                copyContent(x, y);
        }
}
```

Output:

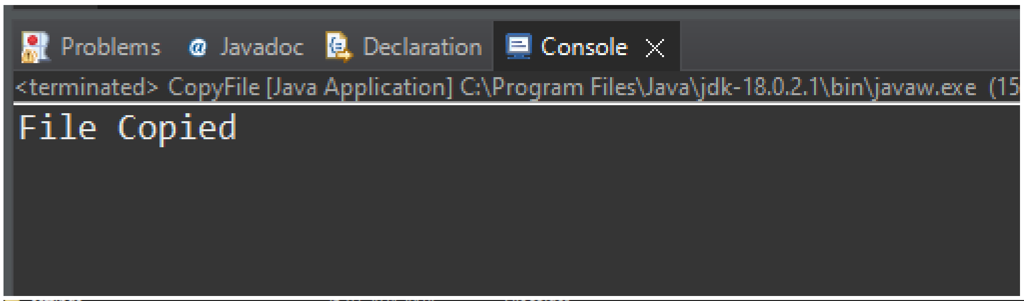


source - Notepad

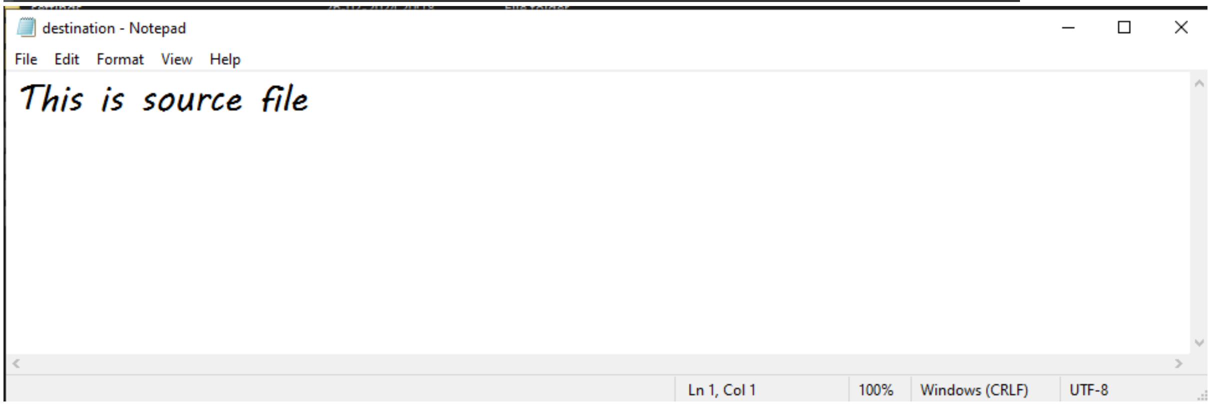
File Edit Format View Help

```
This is source file
```

After copying



```
<terminated> CopyFile [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (15
File Copied
```



destination - Notepad

File Edit Format View Help

```
This is source file
```

## L.6 Develop the java ServerSocket and Socket based TCP/IP client server program and from client side send data records and and on server side store the data records.

Code:

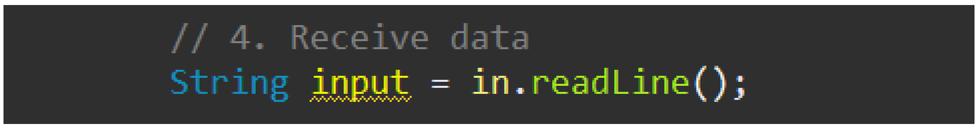
<u>SERVER SIDE</u>

```
package socket;

import java.net.*;

import java.io.*;


public class ServerOne {


        public static void main(String[] args) throws IOException {

                // TODO Auto-generated method stub

                // 1. Create serversocket object

                ServerSocket servSocket = new ServerSocket(1234);


                // 2. Put server into waiting state

                Socket link = servSocket.accept();


                // 3. Set up IO streams

                BufferedReader in = new BufferedReader(new
InputStreamReader(link.getInputStream()));


                // 4. Receive data

                String input = in.readLine();


                // 5. Close connection

                link.close();

        }

}
```

<u>CLIENT SIDE</u>

```
package socket;

import java.net.*;

import java.io.*;


public class ClientOne {

        public static void main(String[] args) throws IOException {

                Socket link = new Socket("localhost", 1234);


                PrintWriter out = new PrintWriter(link.getOutputStream(), true);


                out.println("hello world");


                link.close();

        }

}
```

Output:



```
        // 4. Receive data
        String input = in.readLine();
```

In the server side the data record sent through the output stream i.e., “Hello world” is stored in input variable.

## L.7 Develop the Datagram socket based UDP connection less program and send the String from client side and reverse it on server side and displayed reverse String on client side.

Code:

<u>SERVER SIDE</u>

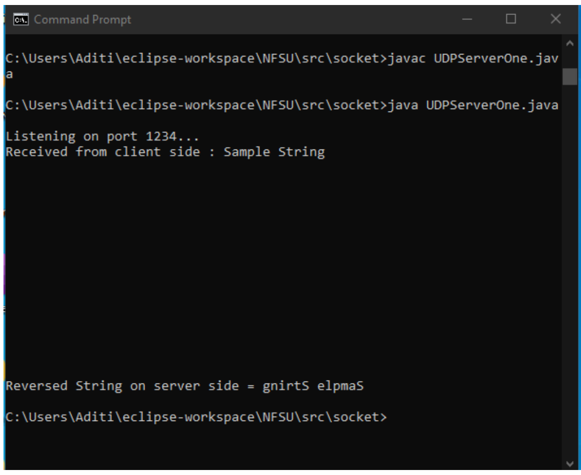
```
package socket;

import java.net.*;

import java.io.*;

public class UDPServerOne {

        public static void main(String[] args) throws IOException{

                // create server side socket

                DatagramSocket serverSocket = new DatagramSocket(1234);

                System.out.println("Listening on port 1234...");


                // set byte array for sending and receiving data

                byte[] receiveData = new byte[1024];

                byte[] sendData = new byte[1024];


                // prepare datagram packet for receiving data from client

                DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);


                // receive data from client side

                serverSocket.receive(receivePacket);

                String sentence = new String(receivePacket.getData());

                System.out.println("Received from client side : "+sentence);


                // set data for packet preperation for sending data to client side

                InetAddress IPAddress = receivePacket.getAddress();

                int Port = receivePacket.getPort();

                // reverse the string received
                char ch[]=sentence.toCharArray();
           String revSentence="";
           for(int i=ch.length-1;i>=0;i--){
              revSentence+=ch[i];
           }
           revSentence = revSentence.trim();
                System.out.println("Reversed String on server side = "+revSentence);
                sendData = revSentence.getBytes();

                // prepare datagram packet to send
                DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,
IPAddress, Port);

                // send packet to client
                serverSocket.send(sendPacket);

                // close
                serverSocket.close();
        }
}
```

CLIENT SIDE

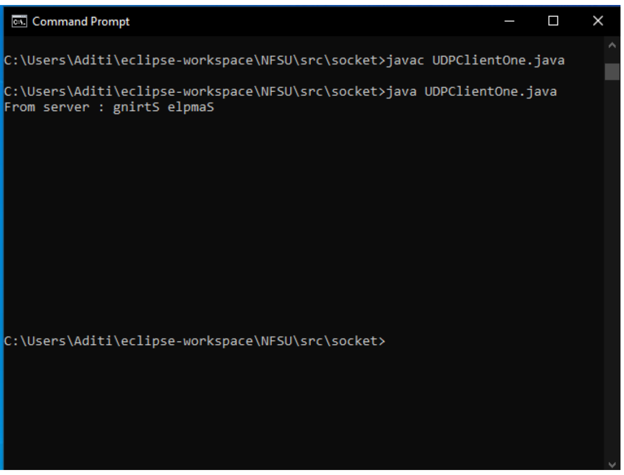
```
package socket;
import java.net.*;
import java.io.*;
public class UDPClientOne {
        public static void main(String[] args) throws IOException{
                // creating datagram socket for client;
                DatagramSocket clientSocket = new DatagramSocket();

                // collecting data to be sent in the datagram packet to send
```

```
                    InetAddress IPAddress = InetAddress.getByName("localhost");
                    int Port = 1234;
                    byte[] sendData = new byte[1024];
                    byte[] receiveData = new byte[1024];
                    String sentence = "Sample String";
                    sendData = sentence.getBytes();

                    // create DatagramPacket and send the data
                    DatagramPacket sendPacket = new DatagramPacket(sendData, sendData.length,
IPAddress, Port);

                    // send the packet
                    clientSocket.send(sendPacket);

                    // create packet to receive data from server
                    DatagramPacket receivePacket = new DatagramPacket(receiveData,
receiveData.length);

                    // receive data from server in receivePacket
                    clientSocket.receive(receivePacket);

                    // store the data received in client side in variable
                    String modifiedSentence = new String(receivePacket.getData());
                    System.out.println("From server : "+modifiedSentence);

                    // close client socket
                    clientSocket.close();
          }
}
```

Output:



```
C:\Users\Aditi\eclipse-workspace\NFSU\src\socket>javac UDPServerOne.jav
a

C:\Users\Aditi\eclipse-workspace\NFSU\src\socket>java UDPServerOne.java

Listening on port 1234...
Received from client side : Sample String









Reversed String on server side = gnirtS elpmaS

C:\Users\Aditi\eclipse-workspace\NFSU\src\socket>
```



```
C:\Users\Aditi\eclipse-workspace\NFSU\src\socket>javac UDPClientOne.java

C:\Users\Aditi\eclipse-workspace\NFSU\src\socket>java UDPClientOne.java
From server : gnirtS elpmaS















C:\Users\Aditi\eclipse-workspace\NFSU\src\socket>
```

## L.8 Develop Multi-threaded client server Chat programming using TCP/IP Sockets and provide the chat-room facility for sharing the content among 3 or more chat clients

Code:

SERVER SIDE:

```
import java.io.IOException;
import java.net.*;

public class Server {

  private ServerSocket serverSocket;

  public Server(ServerSocket serverSocket) {
    this.serverSocket = serverSocket;
  }

  public void startServer() {

    try {
      while(!serverSocket.isClosed()) {
        Socket socket = serverSocket.accept();
        System.out.println("A new client has connected!");
        ClientHandler clientHandler = new ClientHandler(socket);

        Thread thread = new Thread(clientHandler);
        thread.start();
      }
    } catch (IOException e) {
      e.printStackTrace();
    }
    }

    public void closeServerSocket() {
        try {
            if (serverSocket != null) {
                serverSocket.close();
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) throws IOException {
        ServerSocket serverSocket = new ServerSocket(1234);
        Server server = new Server(serverSocket);
        server.startServer();
    }

}
```

<u>CLIENTHANDLER CLASS</u>

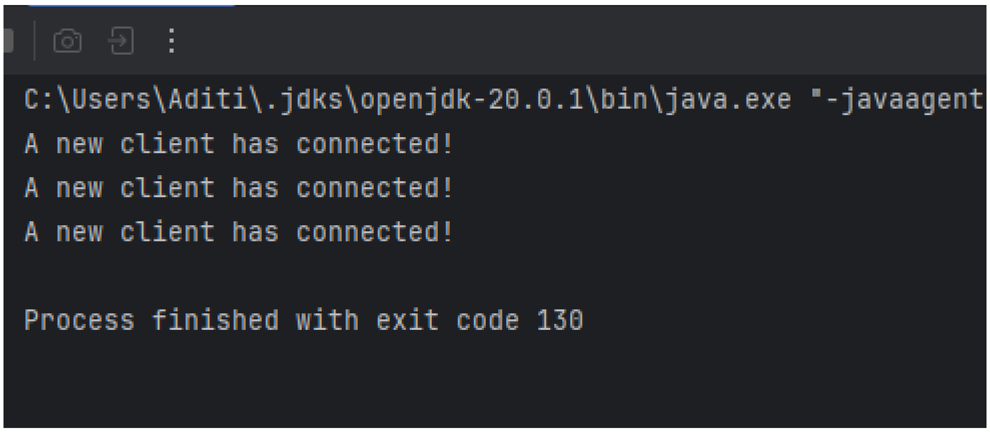
```
import java.io.*;
import java.net.*;
import java.util.*;

public class ClientHandler implements Runnable{

    public static ArrayList<ClientHandler> clientHandlers = new ArrayList<>();
    private Socket socket;
    private BufferedReader bufferedReader;
    private BufferedWriter bufferedWriter;
    private String clientUsername;


    public ClientHandler(Socket socket) {
        try {
            this.socket = socket;
            this.bufferedWriter = new BufferedWriter(new
OutputStreamWriter(socket.getOutputStream()));
            this.bufferedReader = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
            this.clientUsername = bufferedReader.readLine();
            clientHandlers.add(this);
            broadcastMessage("SERVER: "+clientUsername + " has entered the chat!");
        } catch (IOException e) {
            closeEverything(socket, bufferedReader, bufferedWriter);
        }
    }


    @Override
    public void run() {
        String messageFromClient;


        while (socket.isConnected( )) {
            try {
                messageFromClient = bufferedReader.readLine();
                broadcastMessage(messageFromClient);
            } catch (IOException e) {
                closeEverything(socket, bufferedReader, bufferedWriter);
                break;
            }
        }
    }
    public void broadcastMessage(String messageToSend) {
        for (ClientHandler clientHandler : clientHandlers) {
            try {
                if (!clientHandler.clientUsername.equals(clientUsername)) {
                    clientHandler.bufferedWriter.write(messageToSend);
                    clientHandler.bufferedWriter.newLine();
                    clientHandler.bufferedWriter.flush();
                }
            } catch (IOException e) {
                closeEverything(socket, bufferedReader, bufferedWriter);
            }
        }
    }

    public void removeClientHandler() {
        clientHandlers.remove(this);
        broadcastMessage("SERVER: "+ clientUsername + " has left the chat!");
    }

    public void closeEverything(Socket socket, BufferedReader bufferedReader, BufferedWriter
bufferedWriter) {
        removeClientHandler();
        try {
            if (bufferedReader != null) {
                bufferedReader.close();
            }
            if (bufferedWriter != null) {
                bufferedWriter.close();
            }
            if (socket != null) {
                socket.close();
        }
      } catch (IOException e) {
        e.printStackTrace();
      }
    }
}
```

CLIENT SIDE

```
import java.io.*;
import java.net.*;
import java.util.Scanner;

public class Client {
  private Socket socket;
  private BufferedReader bufferedReader;
  private BufferedWriter bufferedWriter;
  private String username;

  public Client(Socket socket, String username) {
    try {
      this.socket = socket;
      this.bufferedWriter = new BufferedWriter(new
OutputStreamWriter(socket.getOutputStream()));
      this.bufferedReader = new BufferedReader(new
InputStreamReader(socket.getInputStream()));
      this.username = username;
    } catch (IOException e) {
      closeEverything(socket, bufferedReader, bufferedWriter);
    }
  }

  public void sendMessage() {
    try {
      bufferedWriter.write(username);
      bufferedWriter.newLine();
      bufferedWriter.flush();

      try (Scanner scanner = new Scanner(System.in)) {
          while (socket.isConnected()) {
              String messageToSend = scanner.nextLine();
              bufferedWriter.write(username + ": " + messageToSend);
              bufferedWriter.newLine();
              bufferedWriter.flush();
          }
      }
    } catch (IOException e) {
      closeEverything(socket, bufferedReader, bufferedWriter);
    }
  }

  public void listenForMessage() {
    new Thread(new Runnable() {
      @Override
      public void run() {
          String msgFromGroupChat;

          while (socket.isConnected()) {
              try {
                  msgFromGroupChat = bufferedReader.readLine();
                  System.out.println(msgFromGroupChat);
              } catch (IOException e) {
                  closeEverything(socket, bufferedReader, bufferedWriter);
              }
      }
    }
    }).start();
  }

  public void closeEverything(Socket socket, BufferedReader bufferedReader, BufferedWriter
bufferedWriter) {
    try {
      if (bufferedReader != null) {
          bufferedReader.close();
      }
      if (bufferedWriter != null) {
          bufferedWriter.close();
      }
      if (socket != null) {
          socket.close();
      }
    } catch (IOException e) {
      e.printStackTrace();
    }
  }

  public static void main(String[] args) throws IOException {
    try (Scanner scanner = new Scanner(System.in)) {
      System.out.println("Enter your username for the group chat: ");
      String username = scanner.nextLine();
      Socket socket = new Socket("localhost", 1234);
      Client client = new Client(socket, username);
      client.listenForMessage();
      client.sendMessage();
    }
  }
}
```

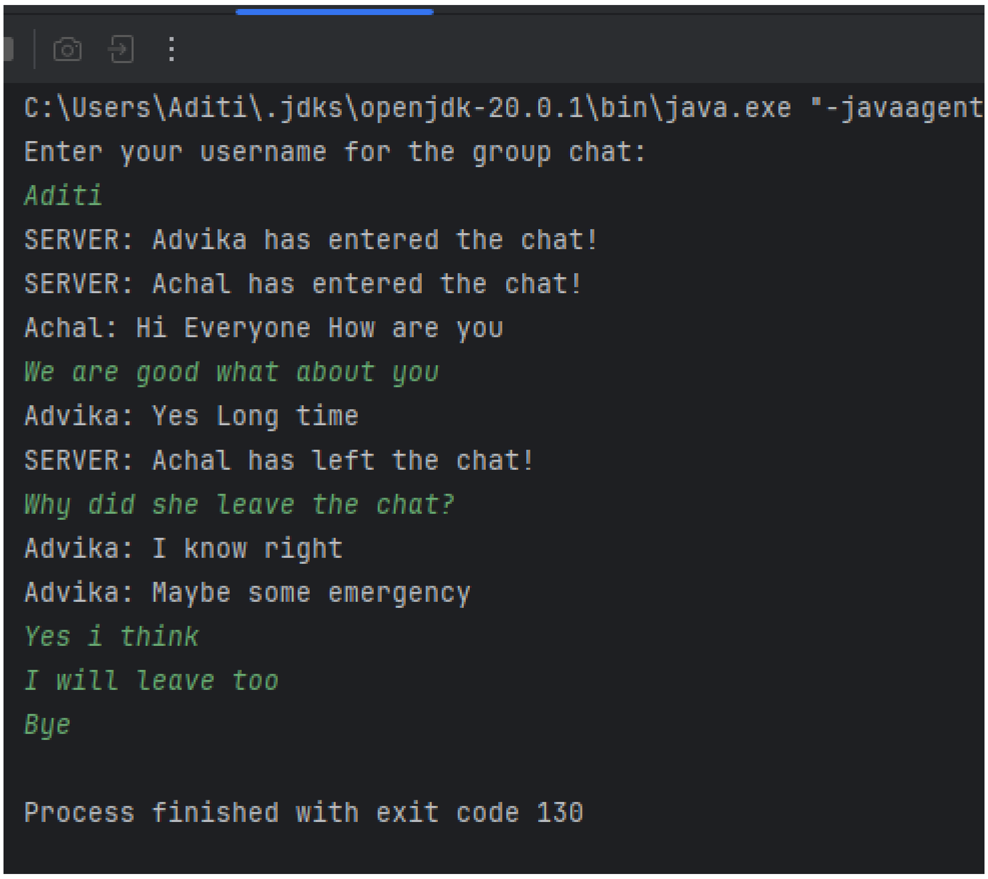
Output:

Server



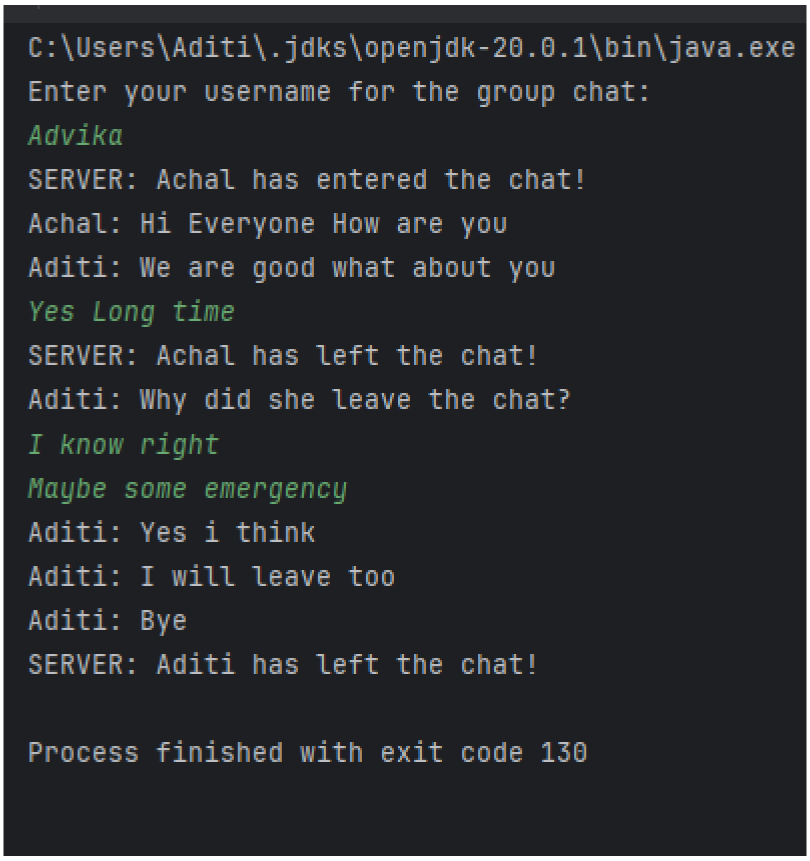
```
C:\Users\Aditi\.jdks\openjdk-20.0.1\bin\java.exe "-javaagent
A new client has connected!
A new client has connected!
A new client has connected!

Process finished with exit code 130
```

Client 1



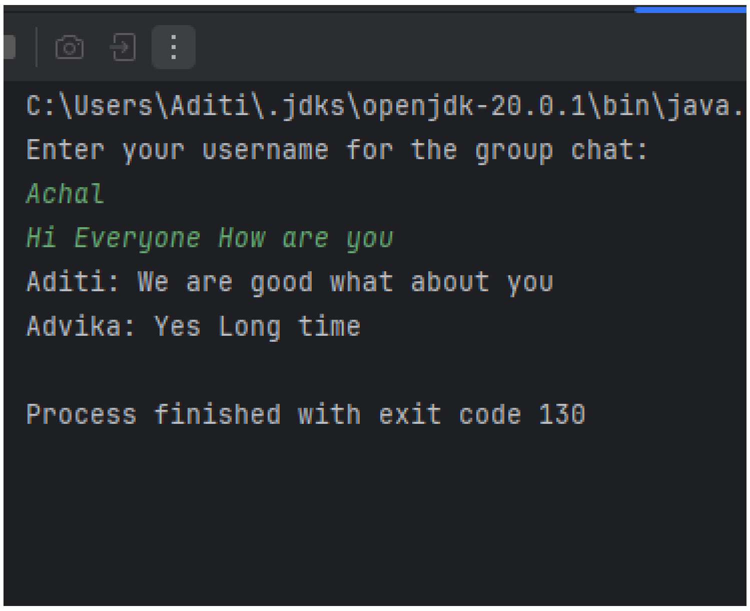
```
C:\Users\Aditi\.jdks\openjdk-20.0.1\bin\java.exe "-javaagent
Enter your username for the group chat:
Aditi
SERVER: Advika has entered the chat!
SERVER: Achal has entered the chat!
Achal: Hi Everyone How are you
We are good what about you
Advika: Yes Long time
SERVER: Achal has left the chat!
Why did she leave the chat?
Advika: I know right
Advika: Maybe some emergency
Yes i think
I will leave too
Bye

Process finished with exit code 130
```

Client 2



```
C:\Users\Aditi\.jdks\openjdk-20.0.1\bin\java.exe
Enter your username for the group chat:
Advika
SERVER: Achal has entered the chat!
Achal: Hi Everyone How are you
Aditi: We are good what about you
Yes Long time
SERVER: Achal has left the chat!
Aditi: Why did she leave the chat?
I know right
Maybe some emergency
Aditi: Yes i think
Aditi: I will leave too
Aditi: Bye
SERVER: Aditi has left the chat!

Process finished with exit code 130
```

Client 3



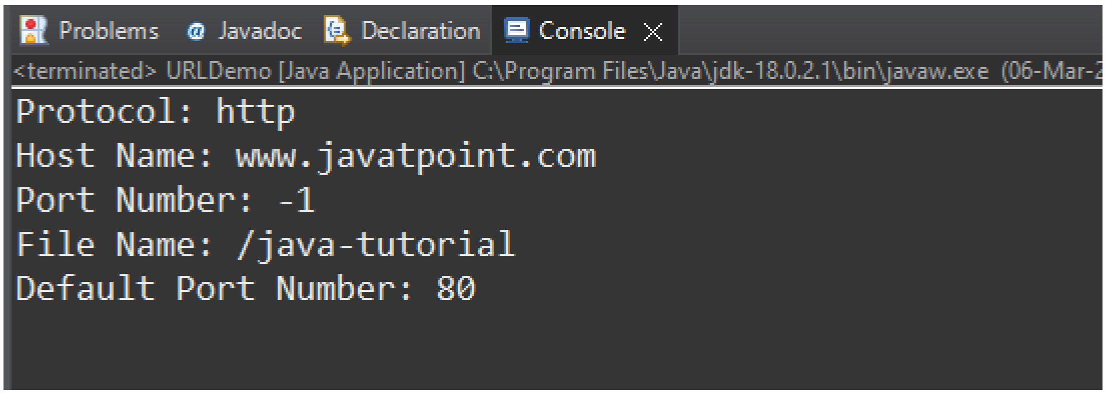
```
C:\Users\Aditi\.jdks\openjdk-20.0.1\bin\java.
Enter your username for the group chat:
Achal
Hi Everyone How are you
Aditi: We are good what about you
Advika: Yes Long time

Process finished with exit code 130
```

## L.9 Develop the java Program using URL and URL connection class and show various URL attributes and develop URLConnection based string to display the any website header attributes and web site content.

Code: URL CLASS

```
package socket;

import java.net.*;

public class URLDemo {

	public static void main(String[] args) {

		// TODO Auto-generated method stub

		try {

			URL url = new URL("http://www.javatpoint.com/java-tutorial");

			System.out.println("Protocol: "+url.getProtocol());

			System.out.println("Host Name: "+url.getHost());

			System.out.println("Port Number: "+url.getPort());

			System.out.println("File Name: "+url.getFile());

			System.out.println("Default Port Number: "+url.getDefaultPort());

		} catch (Exception e) {

			System.out.println(e);

		}

	}

}
```

Output:



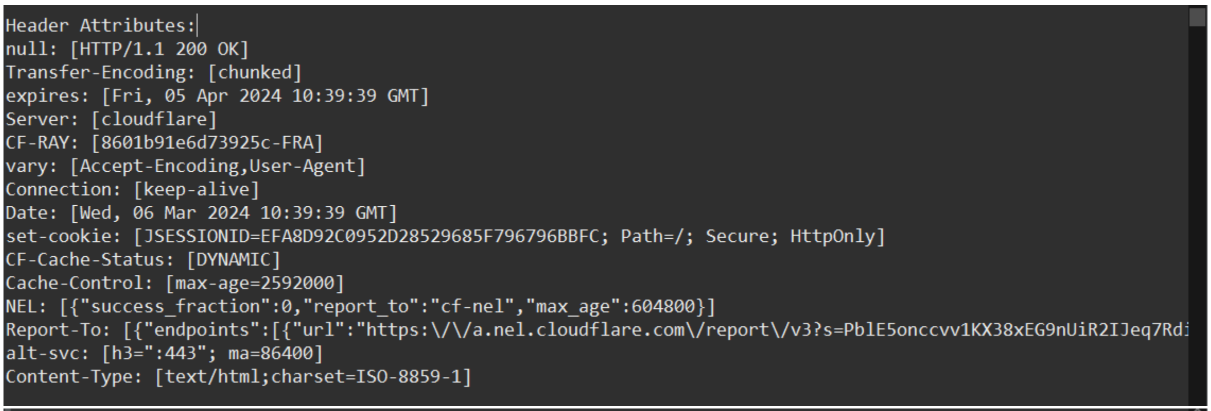
```
<terminated> URLDemo [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (06-Mar-2
Protocol: http
Host Name: www.javatpoint.com
Port Number: -1
File Name: /java-tutorial
Default Port Number: 80
```

Code: URLConnection CLASS

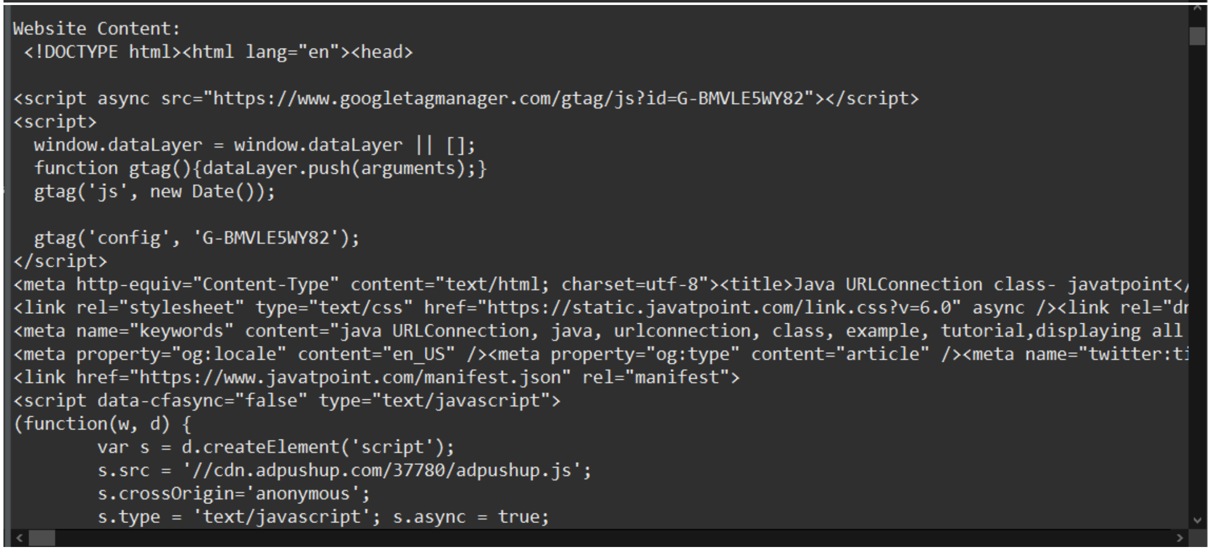
```
package socket;

import java.net.*;

import java.io.*;


public class URLConnectionExample {


        public static void main(String[] args) {

                // TODO Auto-generated method stub

                try {

                        URL url = new URL("https://www.javatpoint.com/URLConnection-class");

                        URLConnection urlcon = url.openConnection();

//                      Display header attributes

       System.out.println("\nHeader Attributes:");

       urlcon.getHeaderFields().forEach((key, value) -> {

         System.out.println(key + ": " + value);

       });


//                      Read and display the website content

       System.out.println("\nWebsite Content:");

       BufferedReader reader = new BufferedReader(new
InputStreamReader(urlcon.getInputStream()));

       String line;

       while ((line = reader.readLine()) != null) {

         System.out.println(line);

       }

                }catch (Exception e) {

                        System.out.println(e);

                }

        }
```

}

Output:



```
Header Attributes:
null: [HTTP/1.1 200 OK]
Transfer-Encoding: [chunked]
expires: [Fri, 05 Apr 2024 10:39:39 GMT]
Server: [cloudflare]
CF-RAY: [8601b91e6d73925c-FRA]
vary: [Accept-Encoding,User-Agent]
Connection: [keep-alive]
Date: [Wed, 06 Mar 2024 10:39:39 GMT]
set-cookie: [JSESSIONID=EFA8D92C0952D28529685F796796BBFC; Path=/; Secure; HttpOnly]
CF-Cache-Status: [DYNAMIC]
Cache-Control: [max-age=2592000]
NEL: [{"success_fraction":0,"report_to":"cf-nel","max_age":604800}]
Report-To: [{"endpoints":[{"url":"https:\/\/a.nel.cloudflare.com\/report\/v3?s=PblE5onccvv1KX38xEG9nUiR2IJeq7Rdi
alt-svc: [h3=":443"; ma=86400]
Content-Type: [text/html;charset=ISO-8859-1]
```



```
Website Content:
 <!DOCTYPE html><html lang="en"><head>

<script async src="https://www.googletagmanager.com/gtag/js?id=G-BMVLE5WY82"></script>
<script>
  window.dataLayer = window.dataLayer || [];
  function gtag(){dataLayer.push(arguments);}
  gtag('js', new Date());

  gtag('config', 'G-BMVLE5WY82');
</script>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"><title>Java URLConnection class- javatpoint</
<link rel="stylesheet" type="text/css" href="https://static.javatpoint.com/link.css?v=6.0" async /><link rel="dr
<meta name="keywords" content="java URLConnection, java, urlconnection, class, example, tutorial,displaying all
<meta property="og:locale" content="en_US" /><meta property="og:type" content="article" /><meta name="twitter:ti
<link href="https://www.javatpoint.com/manifest.json" rel="manifest">
<script data-cfasync="false" type="text/javascript">
(function(w, d) {
        var s = d.createElement('script');
        s.src = '//cdn.adpushup.com/37780/adpushup.js';
        s.crossOrigin='anonymous';
        s.type = 'text/javascript'; s.async = true;
```

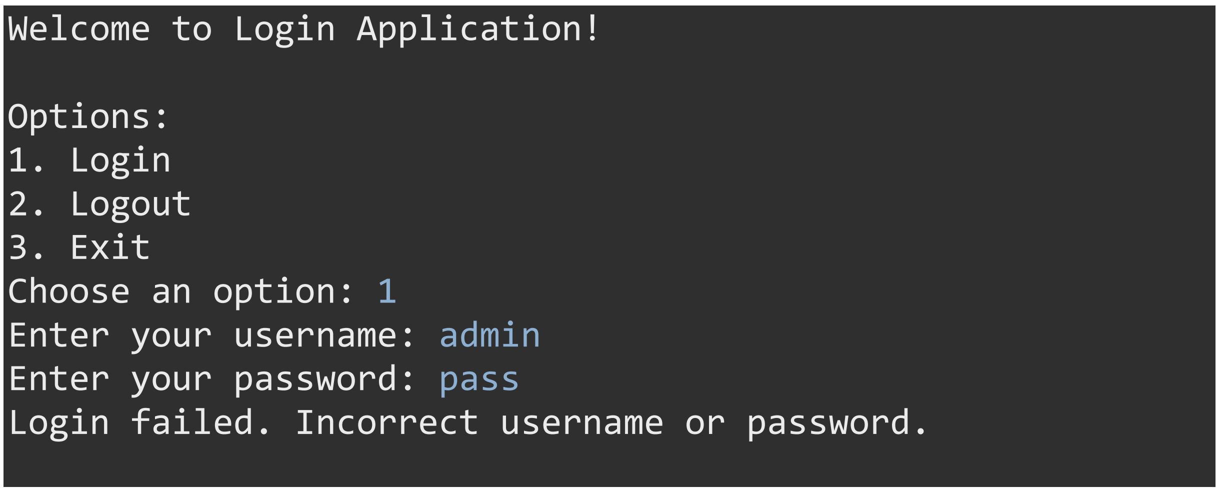
## L.10 Develop JDBC based login authentication system where store the users and password data in MySQL database. Show both the successful and failure login use case.

Code:

```
import java.sql.*;

import java.util.Scanner;


public class LoginApplication{


        private static boolean rememberMe = false;

  private static String loggedInUsername = "";


        public static boolean login(String uname, String pass) throws Exception {


                Class.forName("com.mysql.cj.jdbc.Driver");


                Connection con = DriverManager.getConnection(

                        "jdbc:mysql://localhost:3306/nfsu_dbms?useSSL=false","root","0000");


                Statement stmt=con.createStatement();


                ResultSet rs=stmt.executeQuery("select * from login");


                while(rs.next())  {

                        if (rs.getString(1).equals(uname) & rs.getString(2).endsWith(pass)) {

                                return true;

                        }

                }


                stmt.close();

                con.close();
                return false;


        }

        public static void main(String[] args) throws Exception {

    Scanner scanner = new Scanner(System.in);

    System.out.println("Welcome to Login Application!");

    while (true) {
      System.out.println("\nOptions:");
      System.out.println("1. Login");
      System.out.println("2. Logout");
      System.out.println("3. Exit");
      System.out.print("Choose an option: ");
      int choice = scanner.nextInt();
      scanner.nextLine(); // Consume newline

      switch (choice) {
        case 1:
          login(scanner);
          break;
        case 2:
          logout();
          break;
        case 3:
                System.out.println("Exiting...");
          scanner.close();
          System.exit(0);
```

```
            default:
                System.out.println("Invalid option. Please try again.");
        }
    }
}

// Method to handle login
private static void login(Scanner scanner) throws Exception {
    if (!loggedInUsername.isEmpty()) {
        System.out.println("You are already logged in as " + loggedInUsername);
        return;
    }

    System.out.print("Enter your username: ");
    String enteredUsername = scanner.nextLine();
    System.out.print("Enter your password: ");
    String enteredPassword = scanner.nextLine();

    // Validate credentials
    if (login(enteredUsername, enteredPassword)) {
        loggedInUsername = enteredUsername;
        System.out.println("Login successful!");

        // Remember login if requested
        System.out.print("Do you want to remember your login? (yes/no): ");
        String rememberChoice = scanner.nextLine();
        if (rememberChoice.equalsIgnoreCase("yes")) {
            rememberMe = true;
        } else {
            rememberMe = false;
        }
        } else {
            System.out.println("Login failed. Incorrect username or password.");
        }
    }


    // Method to handle logout
    private static void logout() {
        if (!loggedInUsername.isEmpty()) {
            System.out.println("Logging out user: " + loggedInUsername);
            if (!rememberMe) {
                loggedInUsername = "";
            }
        } else {
            System.out.println("You are not currently logged in.");
        }
    }



}
```

Output:



```
Welcome to Login Application!

Options:
1. Login
2. Logout
3. Exit
Choose an option: 1
Enter your username: admin
Enter your password: pass
Login failed. Incorrect username or password.
```

```
Options:
1. Login
2. Logout
3. Exit
Choose an option: 1
Enter your username: admin
Enter your password: password
Login successful!
Do you want to remember your login? (yes/no): no

Options:
1. Login
2. Logout
3. Exit
Choose an option: 1
You are already logged in as admin

Options:
1. Login
2. Logout
3. Exit
Choose an option: 2
Logging out user: admin

Options:
1. Login
2. Logout
3. Exit
Choose an option: 1
Enter your username: ram
Enter your password: 123
Login successful!
Do you want to remember your login? (yes/no): no

Options:
1. Login
2. Logout
3. Exit
Choose an option: 1
You are already logged in as ram

Options:
```

```
1. Login
2. Logout
3. Exit
Choose an option: 3
Exiting...
```

## L.11 Develop the Student System for any college class room or university and develop the student and teacher tables in MySQL database and provide search based on classroom search or Teacher name search for showing the students in a classroom or students in classroom or both

Code:

```
import java.sql.*;
import java.util.Scanner;


public class StudentSystem {
   // JDBC URL, username, and password of MySQL server
   private static final String JDBC_URL = "jdbc:mysql://localhost:3306/nfsu_dbms";
   private static final String USERNAME = "root";
   private static final String PASSWORD = "0000";


   // Search students by classroom
   public static void searchStudentsByClassroom(String roomNumber) throws Exception{
      Class.forName("com.mysql.cj.jdbc.Driver");
      try (Connection conn = DriverManager.getConnection(JDBC_URL, USERNAME, PASSWORD)) {
         String sql = "SELECT StudentID, Students.FirstName, Students.LastName, Age, Gender FROM
Students INNER JOIN Classroom ON Students.ClassroomID = Classroom.ClassroomID WHERE
RoomNumber = ?";
         PreparedStatement statement = conn.prepareStatement(sql);
         statement.setString(1, roomNumber);
         ResultSet resultSet = statement.executeQuery();


         System.out.println("Students of class number "+roomNumber+" : ");


         System.out.println("-----------------------------------------------------------------------------------------");
         System.out.printf("%10s %15s %15s %10s %10s", "STUDENT_ID", "FIRST_NAME",
"LAST_NAME", "AGE", "GENDER");
         System.out.println();
         System.out.println("-----------------------------------------------------------------------------------------");
```

```
          while (resultSet.next()) {
            int studentID = resultSet.getInt("StudentID");
             String firstName = resultSet.getString("FirstName");
             String lastName = resultSet.getString("LastName");
             int age = resultSet.getInt("Age");
             String gender = resultSet.getString("Gender");
//              System.out.println(studentID+" "+firstName + " " + lastName+" "+age+" "+gender);
             System.out.format("%10s %15s %15s %10s %10s", studentID, firstName, lastName, age,
gender);
             System.out.println();
          }
          System.out.println("--------------------------------------------------------------------------------------------");
       } catch (SQLException e) {
          e.printStackTrace();
       }
   }



   // Search students by teacher name
   public static void searchStudentsByTeacher(String teacherFirstName, String teacherLastName)
throws Exception {
            Class.forName("com.mysql.cj.jdbc.Driver");
       try (Connection conn = DriverManager.getConnection(JDBC_URL, USERNAME, PASSWORD)) {
          String sql = "SELECT StudentID, Students.FirstName, Students.LastName, Age, Gender FROM
Students INNER JOIN Teachers ON Students.ClassroomID = Teachers.ClassroomID WHERE
Teachers.FirstName = ? or Teachers.LastName = ?";
          PreparedStatement statement = conn.prepareStatement(sql);
          statement.setString(1, teacherFirstName);
          statement.setString(2, teacherLastName);
          ResultSet resultSet = statement.executeQuery();

          System.out.println("Students of class with Class Teacher "+teacherFirstName + " " +
teacherLastName+" : ");
```

```
          System.out.println("--------------------------------------------------------------------------------------------");
          System.out.printf("%10s %15s %15s %10s %10s", "STUDENT_ID", "FIRST_NAME",
"LAST_NAME", "AGE", "GENDER");
          System.out.println();
          System.out.println("--------------------------------------------------------------------------------------------");

          while (resultSet.next()) {
            int studentID = resultSet.getInt("StudentID");
             String firstName = resultSet.getString("FirstName");
             String lastName = resultSet.getString("LastName");
             int age = resultSet.getInt("Age");
             String gender = resultSet.getString("Gender");
//              System.out.println(studentID+" "+firstName + " " + lastName+" "+age+" "+gender);
             System.out.format("%10s %15s %15s %10s %10s", studentID, firstName, lastName, age,
gender);
             System.out.println();
          }
          System.out.println("--------------------------------------------------------------------------------------------");

       } catch (SQLException e) {
          e.printStackTrace();
       }
    }

    // Add new student
    public static void addStudent(String firstName, String lastName, int age, String gender, int
classroomID) {
       try (Connection conn = DriverManager.getConnection(JDBC_URL, USERNAME, PASSWORD)) {
          String sql = "INSERT INTO Students (FirstName, LastName, Age, Gender, ClassroomID) VALUES
(?, ?, ?, ?, ?)";
          PreparedStatement statement = conn.prepareStatement(sql);
```

```
            statement.setString(1, firstName);
            statement.setString(2, lastName);
            statement.setInt(3, age);
            statement.setString(4, gender);
            statement.setInt(5, classroomID);
            int rowsInserted = statement.executeUpdate();
            if (rowsInserted > 0) {
                System.out.println("Student added successfully.");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // Delete student
    public static void deleteStudent(int studentID) {
        try (Connection conn = DriverManager.getConnection(JDBC_URL, USERNAME, PASSWORD)) {
            String sql = "DELETE FROM Students WHERE StudentID=?";
            PreparedStatement statement = conn.prepareStatement(sql);
            statement.setInt(1, studentID);
            int rowsDeleted = statement.executeUpdate();
            if (rowsDeleted > 0) {
                System.out.println("Student deleted successfully.");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    // View student details
    public static void viewStudentDetails(int studentID) {
        try (Connection conn = DriverManager.getConnection(JDBC_URL, USERNAME, PASSWORD)) {
            String sql = "SELECT * FROM Students WHERE StudentID=?";
            PreparedStatement statement = conn.prepareStatement(sql);
            statement.setInt(1, studentID);
            ResultSet resultSet = statement.executeQuery();

            if (resultSet.next()) {
                System.out.println("Student ID: " + resultSet.getInt("StudentID"));
                System.out.println("First Name: " + resultSet.getString("FirstName"));
                System.out.println("Last Name: " + resultSet.getString("LastName"));
                System.out.println("Age: " + resultSet.getInt("Age"));
                System.out.println("Gender: " + resultSet.getString("Gender"));
                System.out.println("Classroom ID: " + resultSet.getInt("ClassroomID"));
            } else {
                System.out.println("Student not found.");
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }

    public static void main(String[] args) throws Exception{
//          addStudent("Joey", "Ray", 20, "Male", 1);
//          deleteStudent(1);
//          viewStudentDetails(1);
//      searchStudentsByClassroom("101");
//      searchStudentsByTeacher("David","Wilson");

            try (Scanner scanner = new Scanner(System.in)) {
                            while (true) {
                                System.out.println("\nStudent System Menu:");
            System.out.println("1. Add Student");
            System.out.println("2. Delete Student");
            System.out.println("3. View Student Details");
            System.out.println("4. Search Students by Classroom");
            System.out.println("5. Search Students by Teacher Name");
            System.out.println("6. Exit");
            System.out.print("Enter your choice: ");

            int choice = scanner.nextInt();
            scanner.nextLine();

            switch (choice) {
                case 1:
                    // Add Student
                      System.out.println("\nAdding a new student:");
System.out.print("Enter first name: ");
String firstName = scanner.nextLine();
System.out.print("Enter last name: ");
String lastName = scanner.nextLine();
System.out.print("Enter age: ");
int age = scanner.nextInt();
scanner.nextLine(); // Consume newline character
System.out.print("Enter gender: ");
String gender = scanner.nextLine();
System.out.print("Enter classroom ID: ");
int classroomID = scanner.nextInt();
scanner.nextLine(); // Consume newline character
addStudent(firstName, lastName, age, gender, classroomID);
break;
                case 2:
                    // Delete Student
```

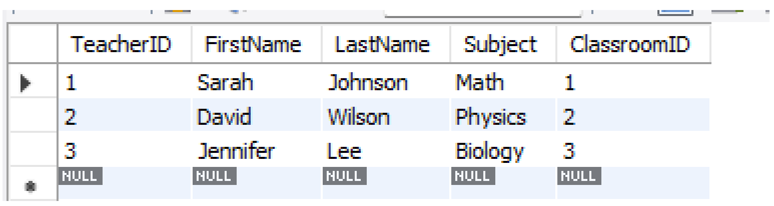
```
          System.out.println("\nDelete a student:");
          System.out.print("Enter student Id to delete: ");
int studentID_del = scanner.nextInt();
scanner.nextLine(); // Consume newline character
deleteStudent(studentID_del);
         break;
       case 3:
         // View Student Details
          System.out.println("\nViewing Student Details:");
          System.out.print("Enter student Id to view details : ");
int studentID_view = scanner.nextInt();
scanner.nextLine(); // Consume newline character
viewStudentDetails(studentID_view);
         break;
       case 4:
         // Search Students by Classroom
          System.out.println("\nSearch Students by Classroom:");
          System.out.print("Enter Classroom Room Number: ");
String room = scanner.nextLine();
searchStudentsByClassroom(room);
         break;
       case 5:
         // Search Students by Teacher Name
          System.out.println("\nSearch Students by Teacher Name:");
          System.out.print("Enter Class Teacher First Name : ");
String teacherfn = scanner.nextLine();
System.out.print("Enter Class Teacher Last Name : ");
String teacherln = scanner.nextLine();
searchStudentsByTeacher(teacherfn, teacherln);
         break;
       case 6:
```

```
                                        // Exit

                                        System.out.println("Exiting...");

                                        System.exit(0);

                                        break;

                                      default:

                                        System.out.println("Invalid choice. Please enter a number between 1
and 6.");

                                    }

                                  }

                         }

   }

}
```

Output:

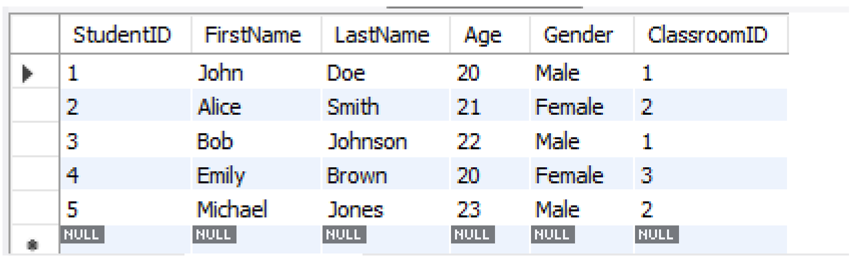
TABLES:

Teachers



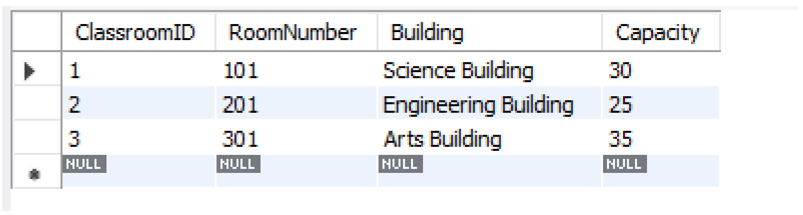
| TeacherID | FirstName | LastName | Subject | ClassroomID |
|---|---|---|---|---|
| 1 | Sarah | Johnson | Math | 1 |
| 2 | David | Wilson | Physics | 2 |
| 3 | Jennifer | Lee | Biology | 3 |
| NULL | NULL | NULL | NULL | NULL |

Students



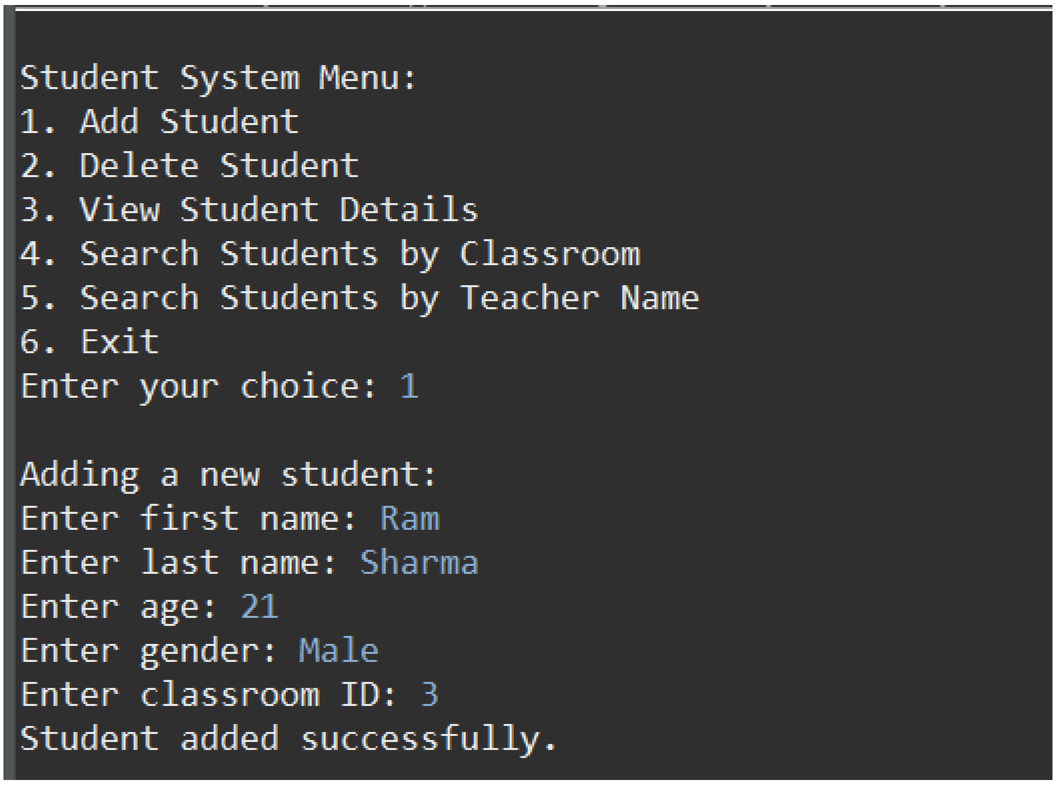
| StudentID | FirstName | LastName | Age | Gender | ClassroomID |
|---|---|---|---|---|---|
| 1 | John | Doe | 20 | Male | 1 |
| 2 | Alice | Smith | 21 | Female | 2 |
| 3 | Bob | Johnson | 22 | Male | 1 |
| 4 | Emily | Brown | 20 | Female | 3 |
| 5 | Michael | Jones | 23 | Male | 2 |
| NULL | NULL | NULL | NULL | NULL | NULL |

Classroom



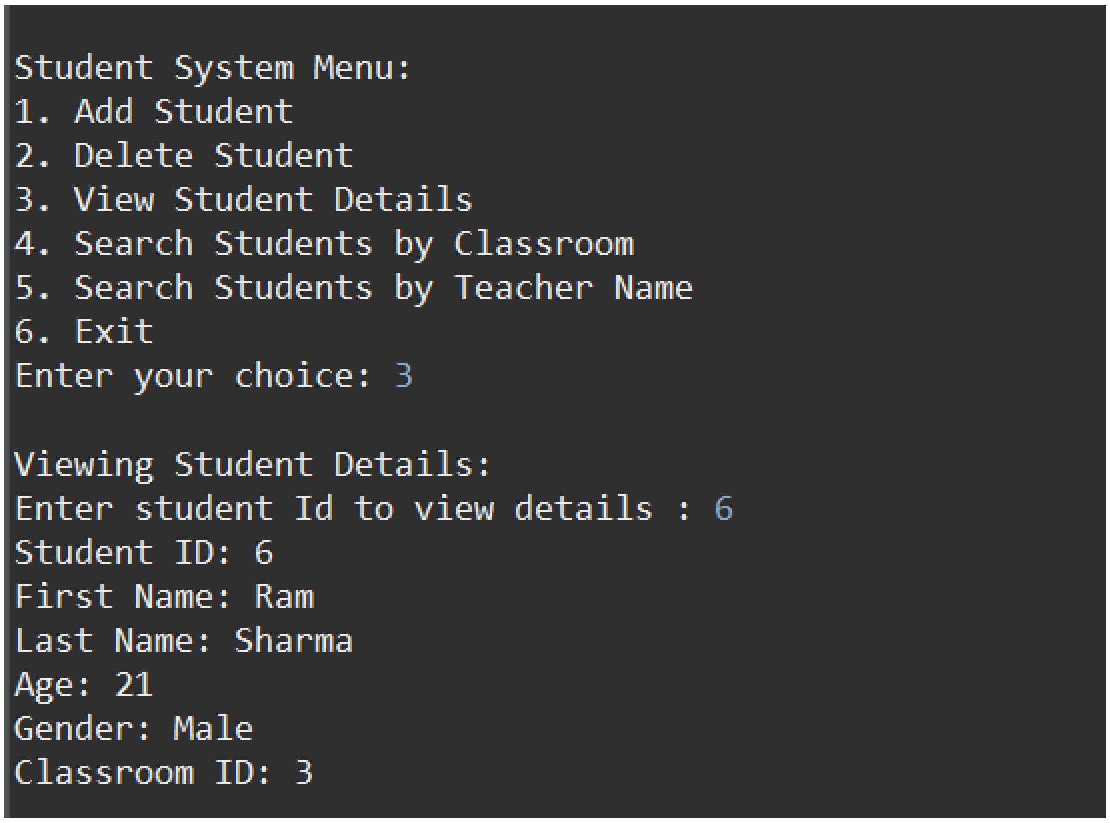
| ClassroomID | RoomNumber | Building | Capacity |
|---|---|---|---|
| 1 | 101 | Science Building | 30 |
| 2 | 201 | Engineering Building | 25 |
| 3 | 301 | Arts Building | 35 |
| NULL | NULL | NULL | NULL |

Adding a student



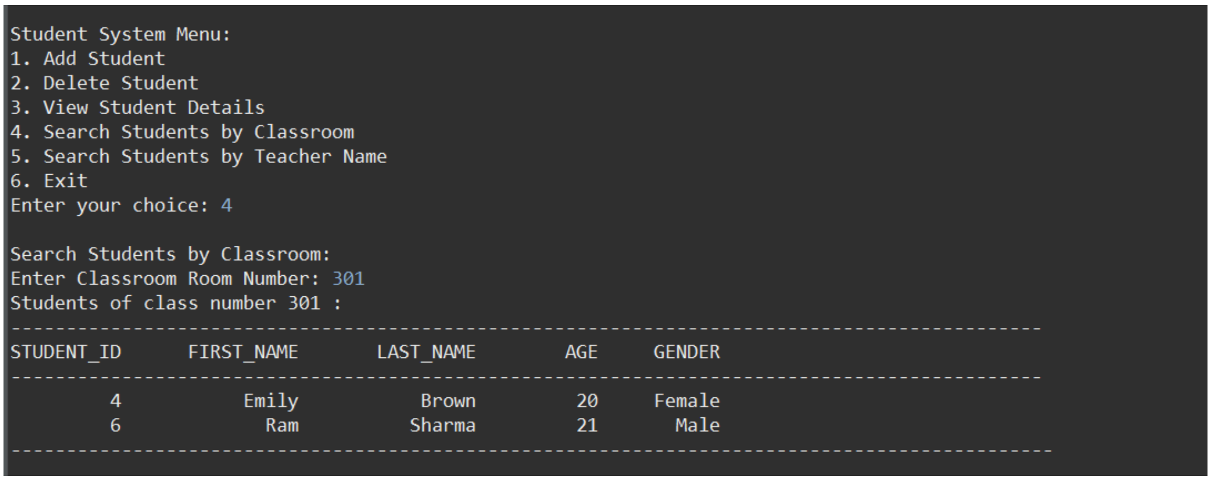
```
Student System Menu:
1. Add Student
2. Delete Student
3. View Student Details
4. Search Students by Classroom
5. Search Students by Teacher Name
6. Exit
Enter your choice: 1

Adding a new student:
Enter first name: Ram
Enter last name: Sharma
Enter age: 21
Enter gender: Male
Enter classroom ID: 3
Student added successfully.
```

Viewing Student Details



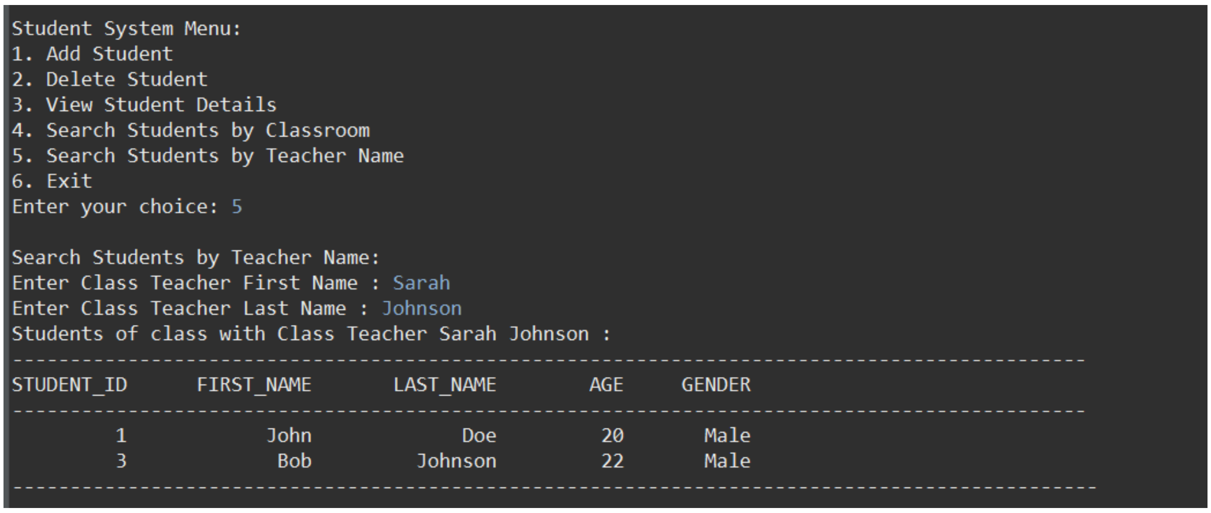
```
Student System Menu:
1. Add Student
2. Delete Student
3. View Student Details
4. Search Students by Classroom
5. Search Students by Teacher Name
6. Exit
Enter your choice: 3

Viewing Student Details:
Enter student Id to view details : 6
Student ID: 6
First Name: Ram
Last Name: Sharma
Age: 21
Gender: Male
Classroom ID: 3
```

Search Students by Classroom



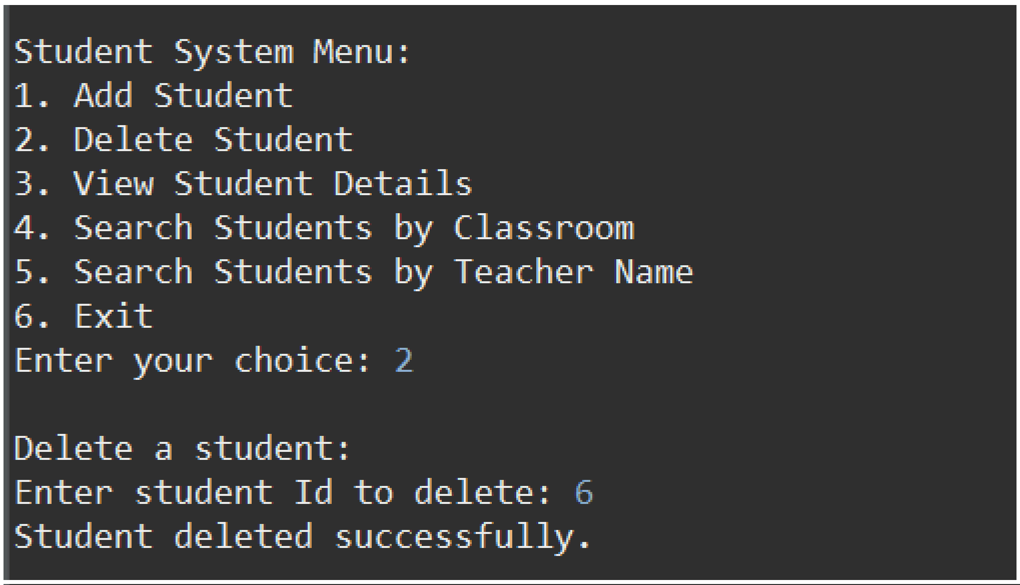
```
Student System Menu:
1. Add Student
2. Delete Student
3. View Student Details
4. Search Students by Classroom
5. Search Students by Teacher Name
6. Exit
Enter your choice: 4

Search Students by Classroom:
Enter Classroom Room Number: 301
Students of class number 301 :
---------------------------------------------------------------------------------------------
STUDENT_ID      FIRST_NAME       LAST_NAME        AGE     GENDER
---------------------------------------------------------------------------------------------
         4           Emily           Brown         20     Female
         6             Ram          Sharma         21       Male
----------------------------------------------------------------------------------------------
```

Search Students by Teacher Name

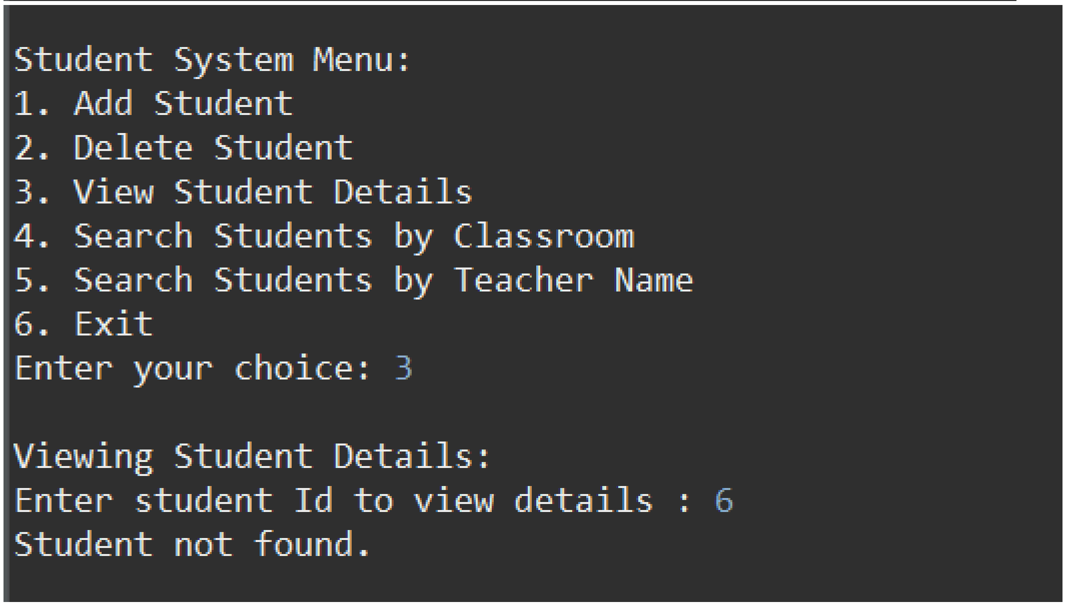


```
Student System Menu:
1. Add Student
2. Delete Student
3. View Student Details
4. Search Students by Classroom
5. Search Students by Teacher Name
6. Exit
Enter your choice: 5

Search Students by Teacher Name:
Enter Class Teacher First Name : Sarah
Enter Class Teacher Last Name : Johnson
Students of class with Class Teacher Sarah Johnson :
---------------------------------------------------------------------------------------------
STUDENT_ID      FIRST_NAME       LAST_NAME        AGE     GENDER
---------------------------------------------------------------------------------------------
         1            John             Doe         20       Male
         3             Bob         Johnson         22       Male
----------------------------------------------------------------------------------------------
```

Delete student.

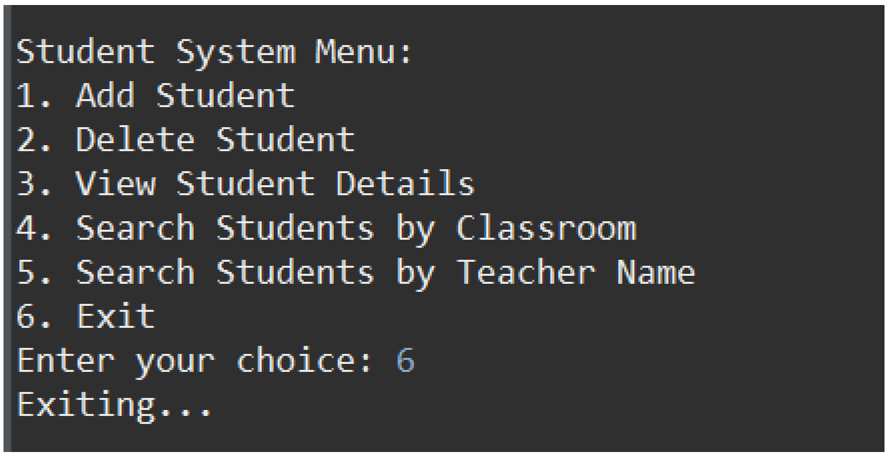


```
Student System Menu:
1. Add Student
2. Delete Student
3. View Student Details
4. Search Students by Classroom
5. Search Students by Teacher Name
6. Exit
Enter your choice: 2

Delete a student:
Enter student Id to delete: 6
Student deleted successfully.
```



```
Student System Menu:
1. Add Student
2. Delete Student
3. View Student Details
4. Search Students by Classroom
5. Search Students by Teacher Name
6. Exit
Enter your choice: 3

Viewing Student Details:
Enter student Id to view details : 6
Student not found.
```

Exit



```
Student System Menu:
1. Add Student
2. Delete Student
3. View Student Details
4. Search Students by Classroom
5. Search Students by Teacher Name
6. Exit
Enter your choice: 6
Exiting...
```

## L.12 Develop Server side Servlet on tomcat Server and for collecting the data from HTML form and store it via Java Servlet in MySQL database.

Code:

index.html

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Student Registration</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f4f4f4;
      margin: 0;
      padding: 0;
    }
    .container {
      width: 400px;
      margin: 50px auto;
      padding: 20px;
      background-color: #fff;
      border-radius: 8px;
      box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
    }
    h2 {
      text-align: center;
      margin-bottom: 20px;
    }
    label {
            display: block;
            margin-bottom: 5px;
            font-weight: bold;
        }
        input[type="text"],
        input[type="number"] {
            width: 100%;
            padding: 10px;
            margin-bottom: 20px;
            border: 1px solid #ccc;
            border-radius: 4px;
            box-sizing: border-box;
        }
        input[type="submit"] {
            width: 100%;
            padding: 10px;
            background-color: #007bff;
            border: none;
            border-radius: 4px;
            color: #fff;
            cursor: pointer;
            transition: background-color 0.3s ease;
        }
        input[type="submit"]:hover {
            background-color: #0056b3;
        }
    </style>
</head>
<body>
    <div class="container">
        <h2>Student Registration</h2>
        <form action="register" method="post">
            <label for="firstName">First Name:</label>
            <input type="text" id="firstName" name="firstName" required>
            <label for="lastName">Last Name:</label>
            <input type="text" id="lastName" name="lastName" required>
            <label for="age">Age:</label>
            <input type="number" id="age" name="age" min="1" required>
            <label for="gender">Gender:</label>
            <input type="text" id="gender" name="gender" required>
            <label for="classroomID">Classroom ID:</label>
            <input type="number" id="classroomID" name="classroomID" min="1" required>
            <input type="submit" value="Register">
        </form>
    </div>
</body>
</html>
```

MyServlet.java

```
package backend;

import java.io.IOException;
import java.io.PrintWriter;
import java.sql.*;

import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

public class MyServlet extends HttpServlet{
	/**
	 *
 */
	private static final long serialVersionUID = 1L;

	public void dbInteraction() throws Exception {

	}

	@Override
	protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
IOException {


		response.setContentType("text/html");
		PrintWriter out = response.getWriter();

		try {
			Class.forName("com.mysql.cj.jdbc.Driver");
		} catch (ClassNotFoundException e) {
			// TODO Auto-generated catch block
			e.printStackTrace();
		}

		String firstName = request.getParameter("firstName");
		String lastName = request.getParameter("lastName");
		String age = request.getParameter("age");
		String gender = request.getParameter("gender");
		String classroomID = request.getParameter("classroomID");

		out.println("<h1>Registration Successful!!</h1>");
		out.println("<br>First Name = "+firstName);
		out.println("<br>Last Name = "+lastName);
```

```
                out.println("<br>Age = "+age);
                out.println("<br>Gender = "+gender);
                out.println("<br>Classroom ID = "+classroomID);

                out.println("<br><br><h2>View Records</h2><br>");

                // JDBC URL, username, and password of MySQL server
           String JDBC_URL = "jdbc:mysql://localhost:3306/nfsu_dbms";
           final String USERNAME = "root";
           final String PASSWORD = "0000";


      try (Connection conn = DriverManager.getConnection(JDBC_URL, USERNAME, PASSWORD)) {

          // Add record
          String sql = "INSERT INTO Students (FirstName, LastName, Age, Gender, ClassroomID)
VALUES (?, ?, ?, ?, ?)";
         PreparedStatement statement = conn.prepareStatement(sql);
         statement.setString(1, firstName);
         statement.setString(2, lastName);
         statement.setInt(3, Integer.parseInt(age));
         statement.setString(4, gender);
         statement.setInt(5, Integer.parseInt(classroomID));
         int rowsInserted = statement.executeUpdate();
         if (rowsInserted > 0) {
           System.out.println("Student added successfully.");
         }


         // View records
         sql = "SELECT StudentID, FirstName, LastName, Age, Gender FROM Students";
```

```
        statement = conn.prepareStatement(sql);

        ResultSet resultSet = statement.executeQuery();


        out.println("<table>");

//          System.out.printf("%10s %15s %15s %10s %10s", "STUDENT_ID", "FIRST_NAME",
"LAST_NAME", "AGE", "GENDER");

        out.println("<tr>"

                + "<th>STUDENT_ID</th>"

                + "<th>FIRST_NAME</th>"

                + "<th>LAST_NAME</th>"

                + "<th>AGE</th>"

                + "<th>GENDER</th>"

                + "</tr>");


//          System.out.println("-----------------------------------------------------------------------------------------
");

        while (resultSet.next()) {

          int studentID = resultSet.getInt("StudentID");

           String fn = resultSet.getString("FirstName");

           String ln = resultSet.getString("LastName");

           int a = resultSet.getInt("Age");

           String g = resultSet.getString("Gender");

//             System.out.println(studentID+" "+firstName + " " + lastName+" "+age+" "+gender);

//             System.out.format("%10s %15s %15s %10s %10s", studentID, fn, ln, a, g);

           out.println("<tr>"

                        + "<td>" + studentID + "</td>"

                        + "<td>" + fn + "</td>"

                        + "<td>" + ln + "</td>"

                        + "<td>" + a + "</td>"

                        + "<td>" + g + "</td>"

                        + "</tr>");

        }
```

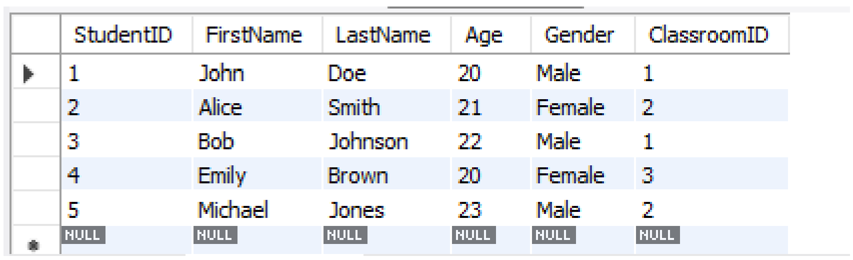
```
          out.println("</table>");
        } catch (SQLException e) {
          e.printStackTrace();
        }


            }


            @Override
            protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
IOException{
                        doGet(request, response);
            }
}
```

<u>web.xml</u>

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://xmlns.jcp.org/xml/ns/javaee"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-
app_4_0.xsd" id="WebApp_ID" version="4.0">
  <servlet>
            <servlet-name>myservlet</servlet-name>
            <servlet-class>backend.MyServlet</servlet-class>
  </servlet>
  <servlet-mapping>
            <servlet-name>myservlet</servlet-name>
            <url-pattern>/register</url-pattern>
  </servlet-mapping>
  <welcome-file-list>
            <welcome-file>index.html</welcome-file>
  </welcome-file-list>
</web-app>
```

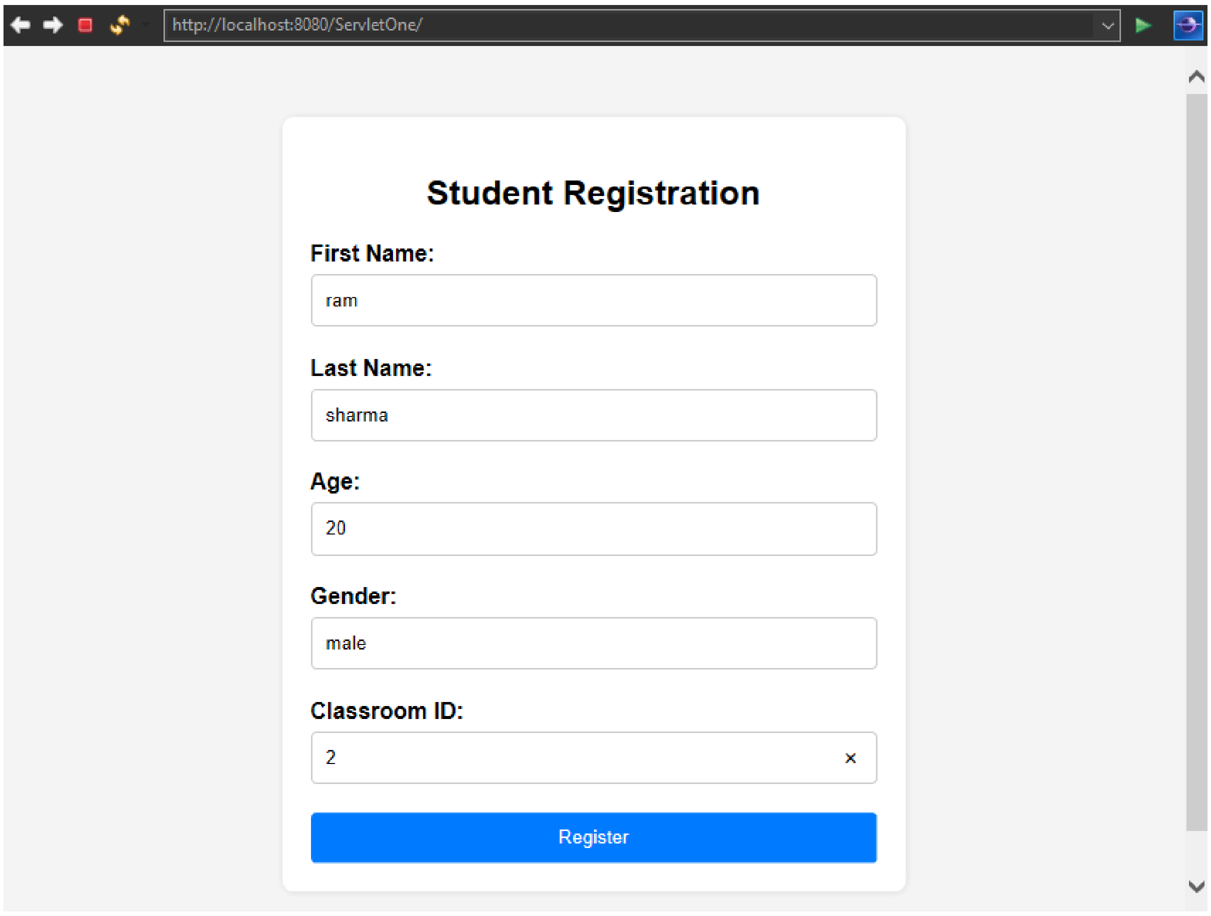
Output:

Student Table



| | StudentID | FirstName | LastName | Age | Gender | ClassroomID |
|---|---|---|---|---|---|---|
| ▶ | 1 | John | Doe | 20 | Male | 1 |
| | 2 | Alice | Smith | 21 | Female | 2 |
| | 3 | Bob | Johnson | 22 | Male | 1 |
| | 4 | Emily | Brown | 20 | Female | 3 |
| | 5 | Michael | Jones | 23 | Male | 2 |
| * | NULL | NULL | NULL | NULL | NULL | NULL |

Index.html => Home Page



http://localhost:8080/ServletOne/

**Student Registration**

First Name: ram

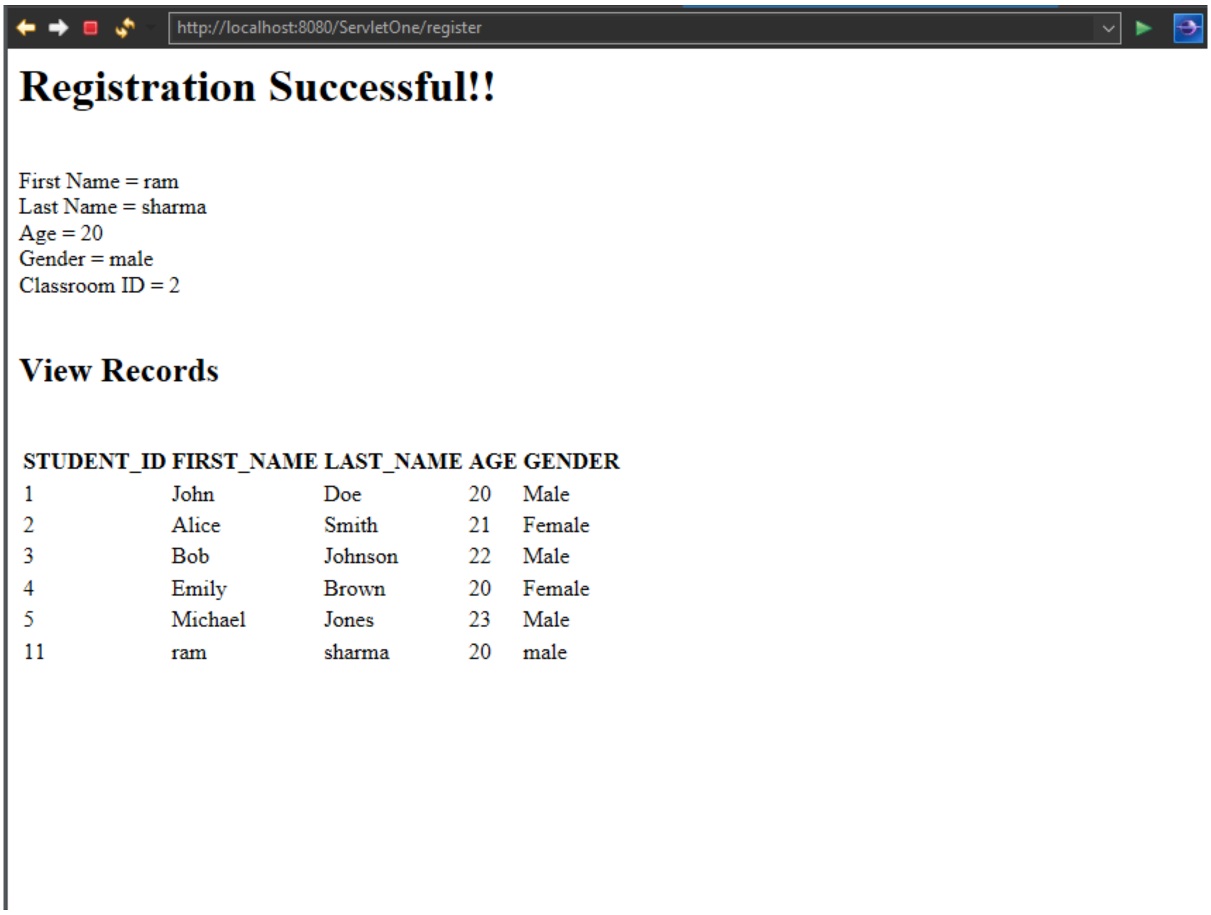
Last Name: sharma

Age: 20

Gender: male

Classroom ID: 2

After registering => /register i.e., MyServlet.java



http://localhost:8080/ServletOne/register

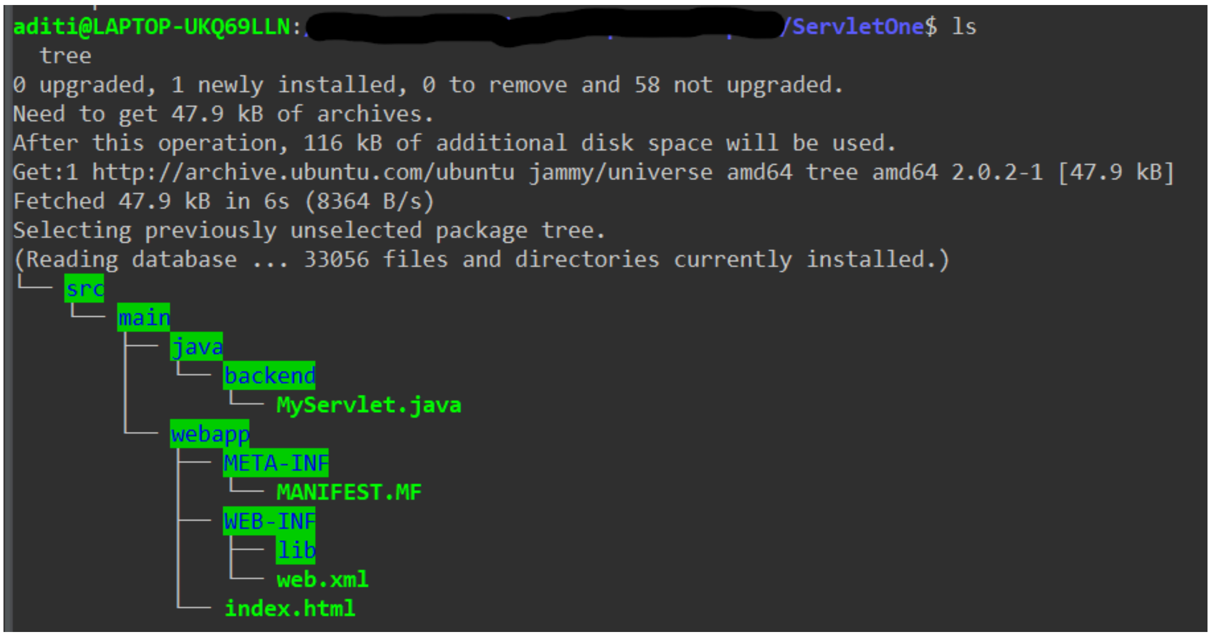
# Registration Successful!!

First Name = ram
Last Name = sharma
Age = 20
Gender = male
Classroom ID = 2

## View Records

| STUDENT_ID | FIRST_NAME | LAST_NAME | AGE | GENDER |
|---|---|---|---|---|
| 1 | John | Doe | 20 | Male |
| 2 | Alice | Smith | 21 | Female |
| 3 | Bob | Johnson | 22 | Male |
| 4 | Emily | Brown | 20 | Female |
| 5 | Michael | Jones | 23 | Male |
| 11 | ram | sharma | 20 | male |

Project Structure



```
aditi@LAPTOP-UKQ69LLN:[redacted]/ServletOne$ ls
  tree
0 upgraded, 1 newly installed, 0 to remove and 58 not upgraded.
Need to get 47.9 kB of archives.
After this operation, 116 kB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu jammy/universe amd64 tree amd64 2.0.2-1 [47.9 kB]
Fetched 47.9 kB in 6s (8364 B/s)
Selecting previously unselected package tree.
(Reading database ... 33056 files and directories currently installed.)
└── src
    └── main
        ├── java
        │   └── backend
        │       └── MyServlet.java
        └── webapp
            ├── META-INF
            │   └── MANIFEST.MF
            ├── WEB-INF
            │   ├── lib
            │   └── web.xml
            └── index.html
```