Computer Networks Lab(CS302)

Assignment-2 Report

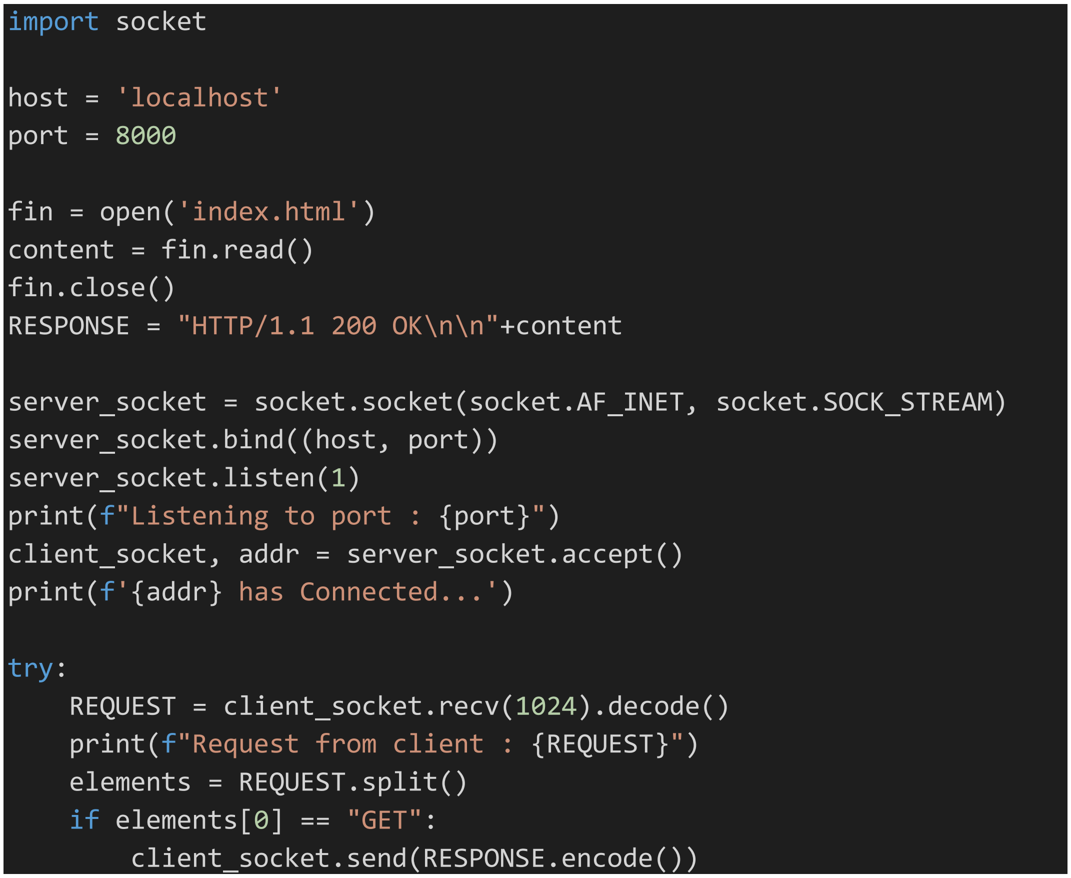
Team Members:-

1. Prajwal J M – 191CS143
2. Saurabh Mokashi – 191CS154

Q1) Using TCP, implement HTTP server and client

a) Code:-

i. Server side



```
import socket

host = 'localhost'
port = 8000

fin = open('index.html')
content = fin.read()
fin.close()
RESPONSE = "HTTP/1.1 200 OK\n\n"+content

server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server_socket.bind((host, port))
server_socket.listen(1)
print(f"Listening to port : {port}")
client_socket, addr = server_socket.accept()
print(f'{addr} has Connected...')

try:
    REQUEST = client_socket.recv(1024).decode()
    print(f"Request from client : {REQUEST}")
    elements = REQUEST.split()
    if elements[0] == "GET":
        client_socket.send(RESPONSE.encode())
    else:
        client_socket.send("HTTP/1.1 400 Bad Request\n\nNot a valid Method".encode())
    client_socket.close()
except:
    client_socket.send("HTTP/1.1 404 Not Found\n\nError Not Found".encode())
    print(f"Connection Closed Abruptly for {addr}")
    client_socket.close()

print("Closing Connection")
server_socket.close()
```

ii. Client side

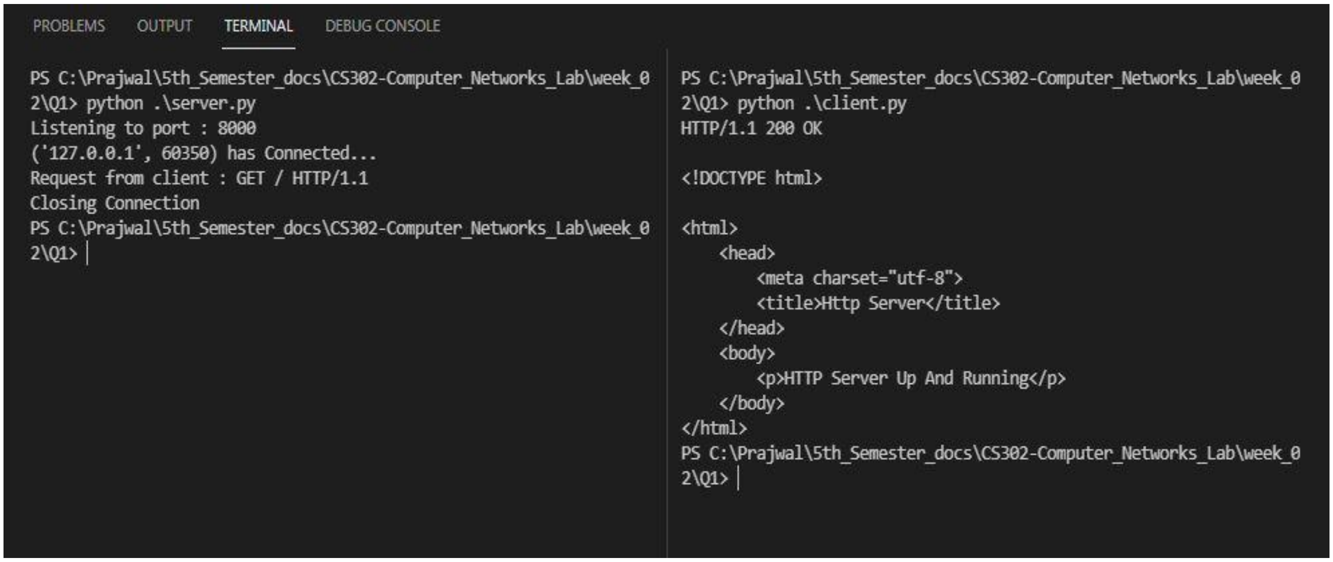
```
import socket

host = '127.0.0.1'
port = 8000

client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
client_socket.connect((host, port))

client_socket.send("GET / HTTP/1.1".encode())

RESPONSE = client_socket.recv(1024).decode()

print(RESPONSE)

client_socket.close()
```

b) Explanation:-

On server side the TCP server socket is created and binded to localhost port 8000. Then the HTTP Response is generated which would be sent on any HTTP GET request from any clients. Now the server socket starts accepting request messages from any clients. Once it receives request message it checks if it is a GET method and returns HTTP response message that was generated

On client side TCP client socket is created and connected to server. Then we send a HTTP request message and get back the HTTP Response and display it on the screen

c) Output:-



```
PS C:\Prajwal\5th_Semester_docs\CS302-Computer_Networks_Lab\week_0
2\Q1> python .\server.py
Listening to port : 8000
('127.0.0.1', 60350) has Connected...
Request from client : GET / HTTP/1.1
Closing Connection
PS C:\Prajwal\5th_Semester_docs\CS302-Computer_Networks_Lab\week_0
2\Q1> |
```

```
PS C:\Prajwal\5th_Semester_docs\CS302-Computer_Networks_Lab\week_0
2\Q1> python .\client.py
HTTP/1.1 200 OK

<!DOCTYPE html>

<html>
    <head>
        <meta charset="utf-8">
        <title>Http Server</title>
    </head>
    <body>
        <p>HTTP Server Up And Running</p>
    </body>
</html>
PS C:\Prajwal\5th_Semester_docs\CS302-Computer_Networks_Lab\week_0
2\Q1> |
```

Q2) Write a program to translate domain name to IP address

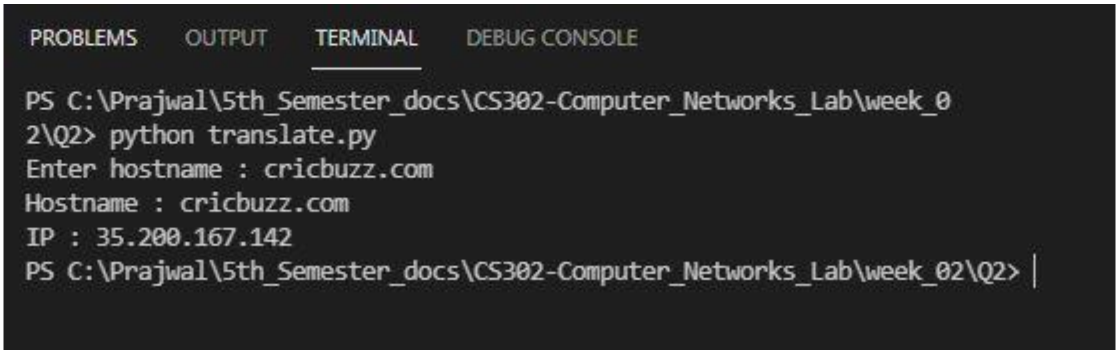
a) Code:-

```python
import socket

hostname = input("Enter hostname : ")
IP = socket.gethostbyname(hostname)
print("Hostname :", hostname)
print("IP :", IP)
```

b) Explanation:-

We create a socket and by using 'gethostbyname' function we get the IP address of the inputted domain name directly.

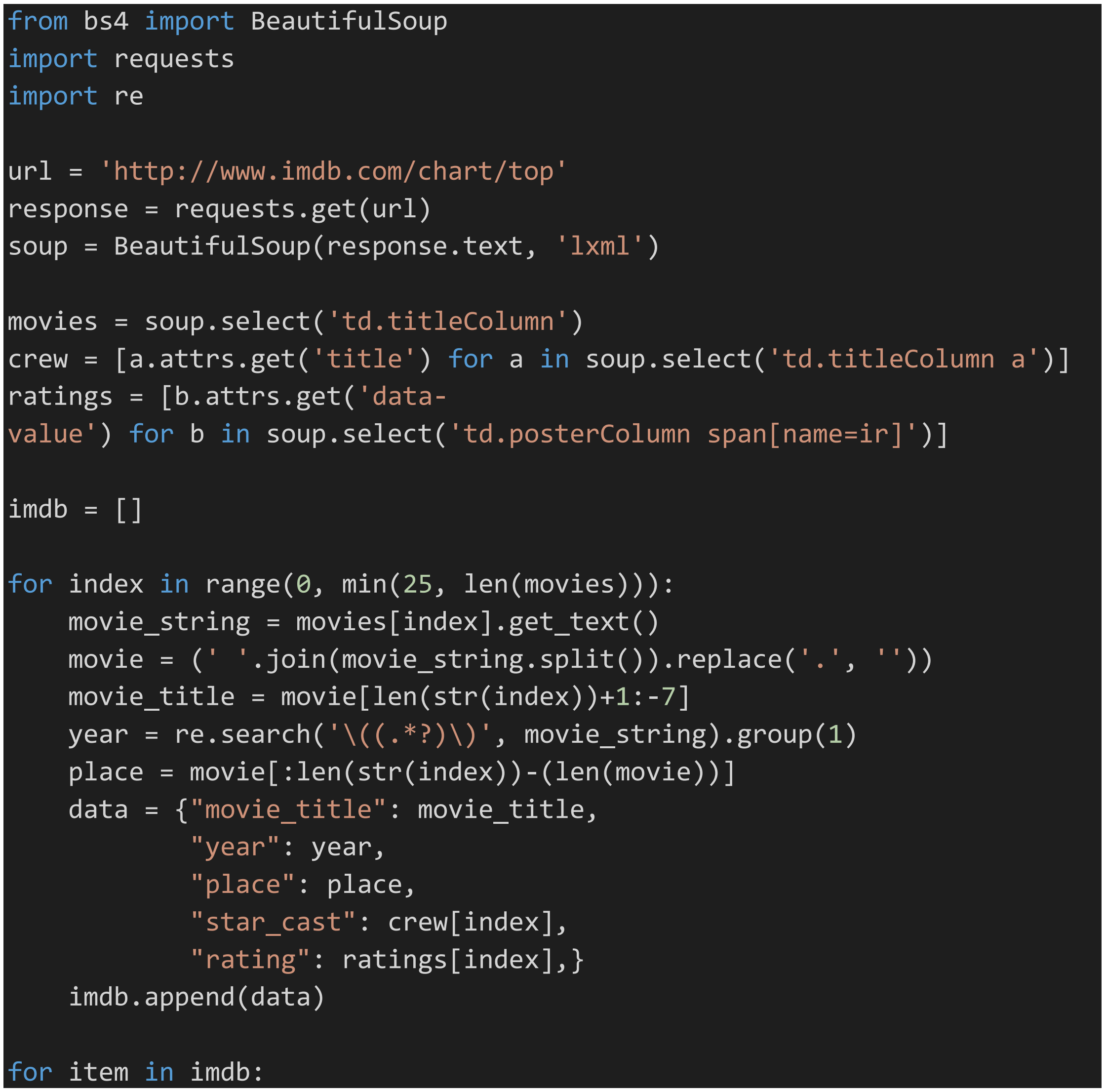
c) Output:-



```
PS C:\Prajwal\5th_Semester_docs\CS302-Computer_Networks_Lab\week_0
2\Q2> python translate.py
Enter hostname : cricbuzz.com
Hostname : cricbuzz.com
IP : 35.200.167.142
PS C:\Prajwal\5th_Semester_docs\CS302-Computer_Networks_Lab\week_02\Q2> |
```

Q3) Develop a program to view data of top movies in IMDB

a) Code:-

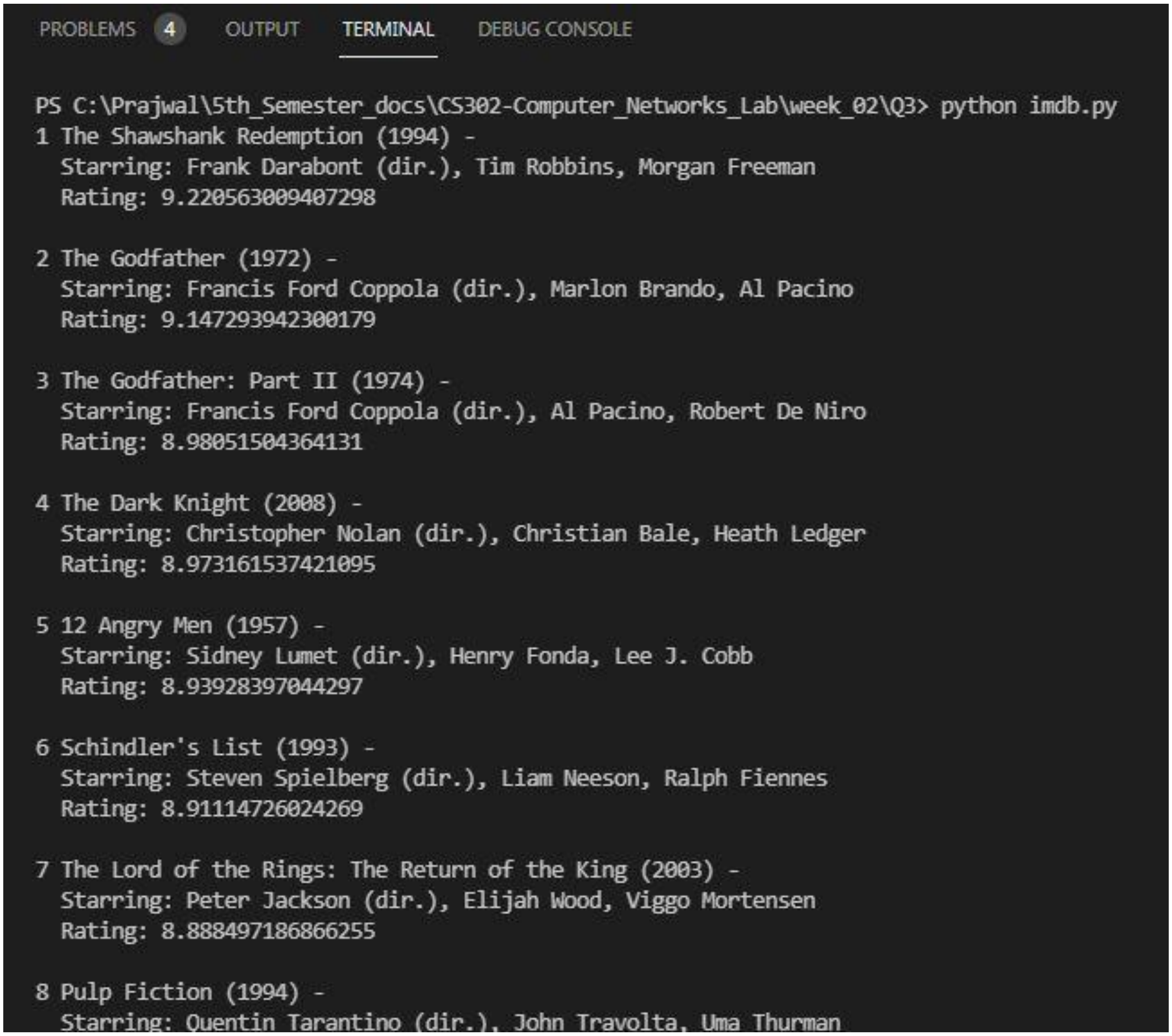


```
from bs4 import BeautifulSoup
import requests
import re

url = 'http://www.imdb.com/chart/top'
response = requests.get(url)
soup = BeautifulSoup(response.text, 'lxml')

movies = soup.select('td.titleColumn')
crew = [a.attrs.get('title') for a in soup.select('td.titleColumn a')]
ratings = [b.attrs.get('data-
value') for b in soup.select('td.posterColumn span[name=ir]')]

imdb = []

for index in range(0, min(25, len(movies))):
    movie_string = movies[index].get_text()
    movie = (' '.join(movie_string.split()).replace('.', ''))
    movie_title = movie[len(str(index))+1:-7]
    year = re.search('\((.*?)\)', movie_string).group(1)
    place = movie[:len(str(index))-(len(movie))]
    data = {"movie_title": movie_title,
            "year": year,
            "place": place,
            "star_cast": crew[index],
            "rating": ratings[index],}
    imdb.append(data)

for item in imdb:
    print(item['place'], item['movie_title'], '('+item['year']+') -
', '\n ' ,'Starring:', item['star_cast'], '\n  '
    'Rating:', item['rating'], '\n  ')
```

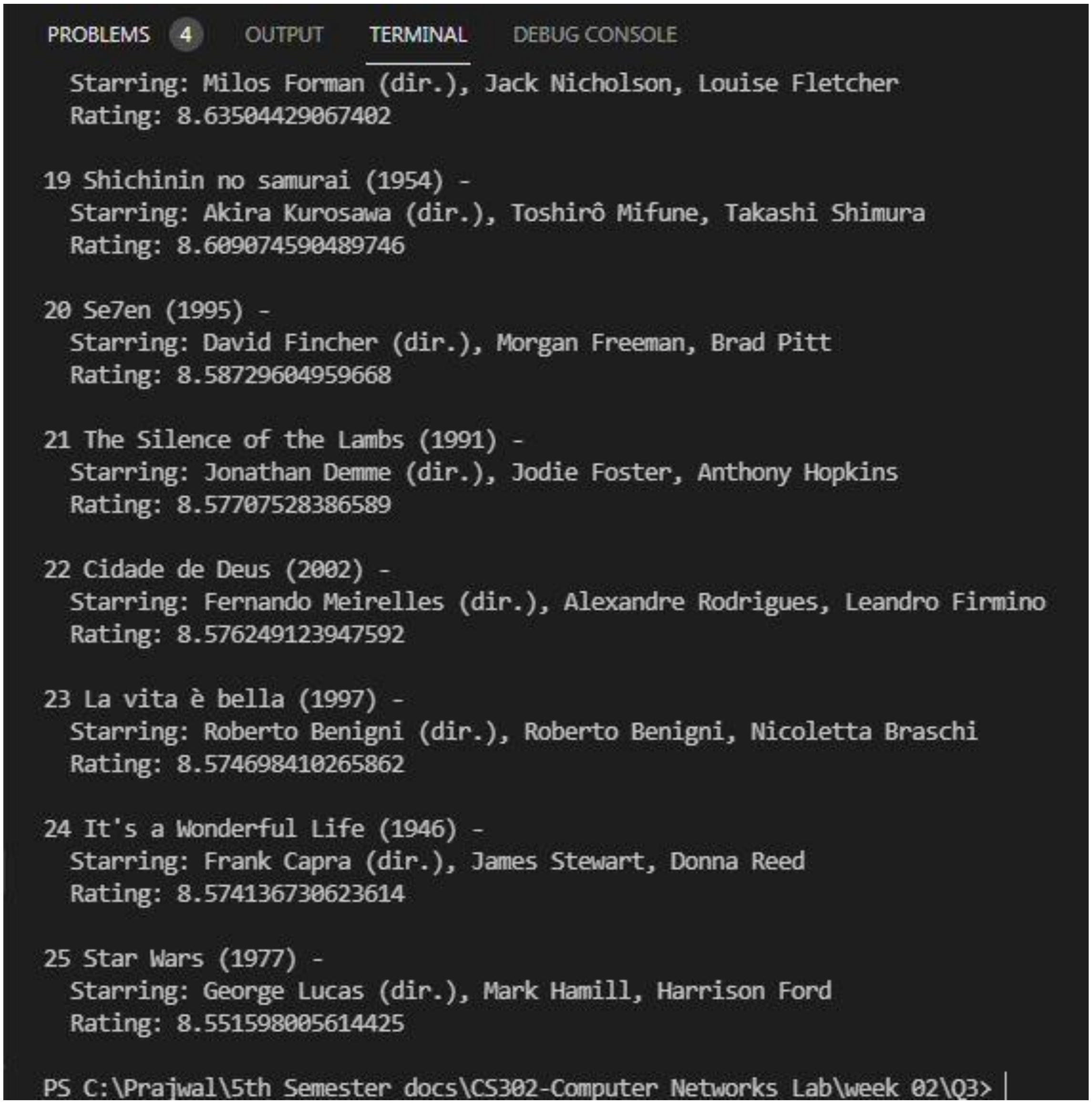
b) Explanation:-

Using ‘requests’ and ‘BeautifulSoup’ packages of python we will be scraping from the website *'http://www.imdb.com/chart/top'* . Then the received data is segregated to get the rank/place, movie title, year, cast and rating of the top 25 IMDb movies and displayed.

c) Output:-



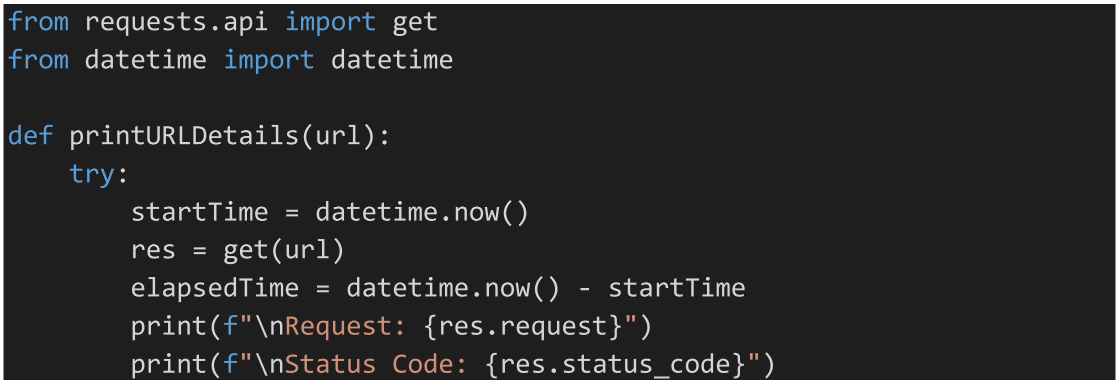
```
PS C:\Prajwal\5th_Semester_docs\CS302-Computer_Networks_Lab\week_02\Q3> python imdb.py
1 The Shawshank Redemption (1994) -
  Starring: Frank Darabont (dir.), Tim Robbins, Morgan Freeman
  Rating: 9.220563009407298

2 The Godfather (1972) -
  Starring: Francis Ford Coppola (dir.), Marlon Brando, Al Pacino
  Rating: 9.147293942300179

3 The Godfather: Part II (1974) -
  Starring: Francis Ford Coppola (dir.), Al Pacino, Robert De Niro
  Rating: 8.98051504364131

4 The Dark Knight (2008) -
  Starring: Christopher Nolan (dir.), Christian Bale, Heath Ledger
  Rating: 8.973161537421095

5 12 Angry Men (1957) -
  Starring: Sidney Lumet (dir.), Henry Fonda, Lee J. Cobb
  Rating: 8.93928397044297

6 Schindler's List (1993) -
  Starring: Steven Spielberg (dir.), Liam Neeson, Ralph Fiennes
  Rating: 8.91114726024269

7 The Lord of the Rings: The Return of the King (2003) -
  Starring: Peter Jackson (dir.), Elijah Wood, Viggo Mortensen
  Rating: 8.888497186866255

8 Pulp Fiction (1994) -
  Starring: Quentin Tarantino (dir.), John Travolta, Uma Thurman
```



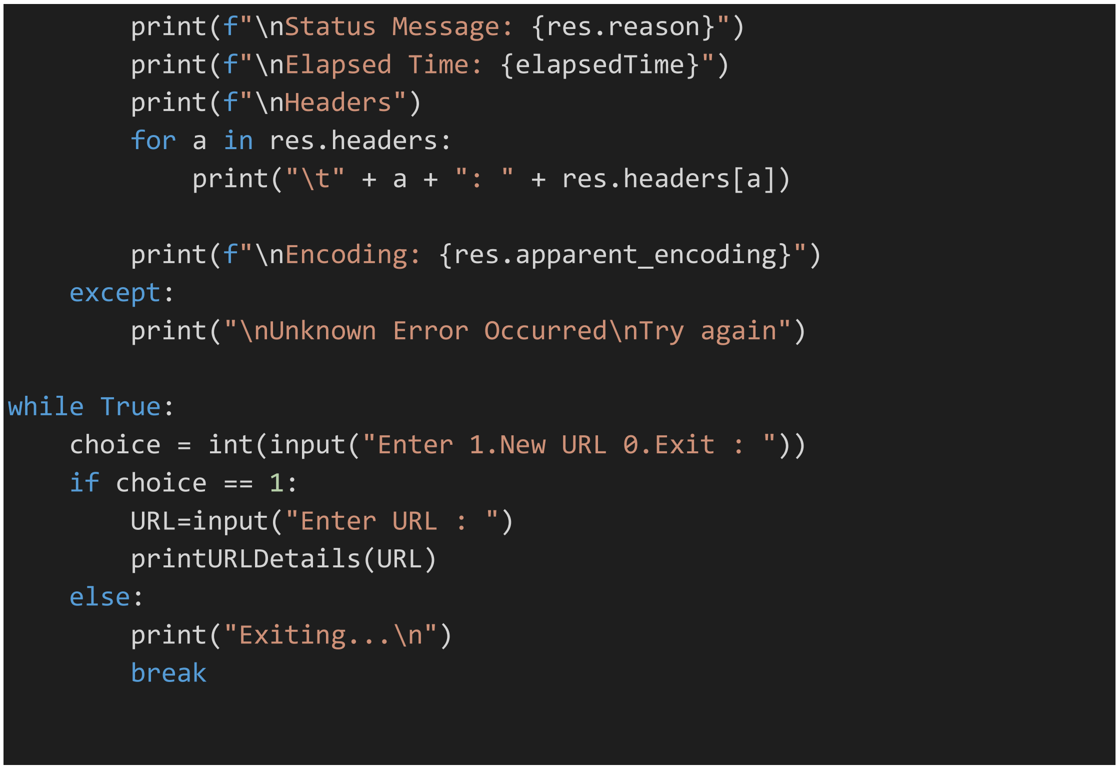
```
  Starring: Milos Forman (dir.), Jack Nicholson, Louise Fletcher
  Rating: 8.63504429067402

19 Shichinin no samurai (1954) -
  Starring: Akira Kurosawa (dir.), Toshirô Mifune, Takashi Shimura
  Rating: 8.609074590489746

20 Se7en (1995) -
  Starring: David Fincher (dir.), Morgan Freeman, Brad Pitt
  Rating: 8.58729604959668

21 The Silence of the Lambs (1991) -
  Starring: Jonathan Demme (dir.), Jodie Foster, Anthony Hopkins
  Rating: 8.57707528386589

22 Cidade de Deus (2002) -
  Starring: Fernando Meirelles (dir.), Alexandre Rodrigues, Leandro Firmino
  Rating: 8.576249123947592

23 La vita è bella (1997) -
  Starring: Roberto Benigni (dir.), Roberto Benigni, Nicoletta Braschi
  Rating: 8.574698410265862

24 It's a Wonderful Life (1946) -
  Starring: Frank Capra (dir.), James Stewart, Donna Reed
  Rating: 8.574136730623614

25 Star Wars (1977) -
  Starring: George Lucas (dir.), Mark Hamill, Harrison Ford
  Rating: 8.551598005614425

PS C:\Prajwal\5th Semester docs\CS302-Computer Networks Lab\week 02\Q3>
```

Q4) Write program to display details of input URL

a) Code:-



```python
from requests.api import get
from datetime import datetime

def printURLDetails(url):
    try:
        startTime = datetime.now()
        res = get(url)
        elapsedTime = datetime.now() - startTime
        print(f"\nRequest: {res.request}")
        print(f"\nStatus Code: {res.status_code}")
```



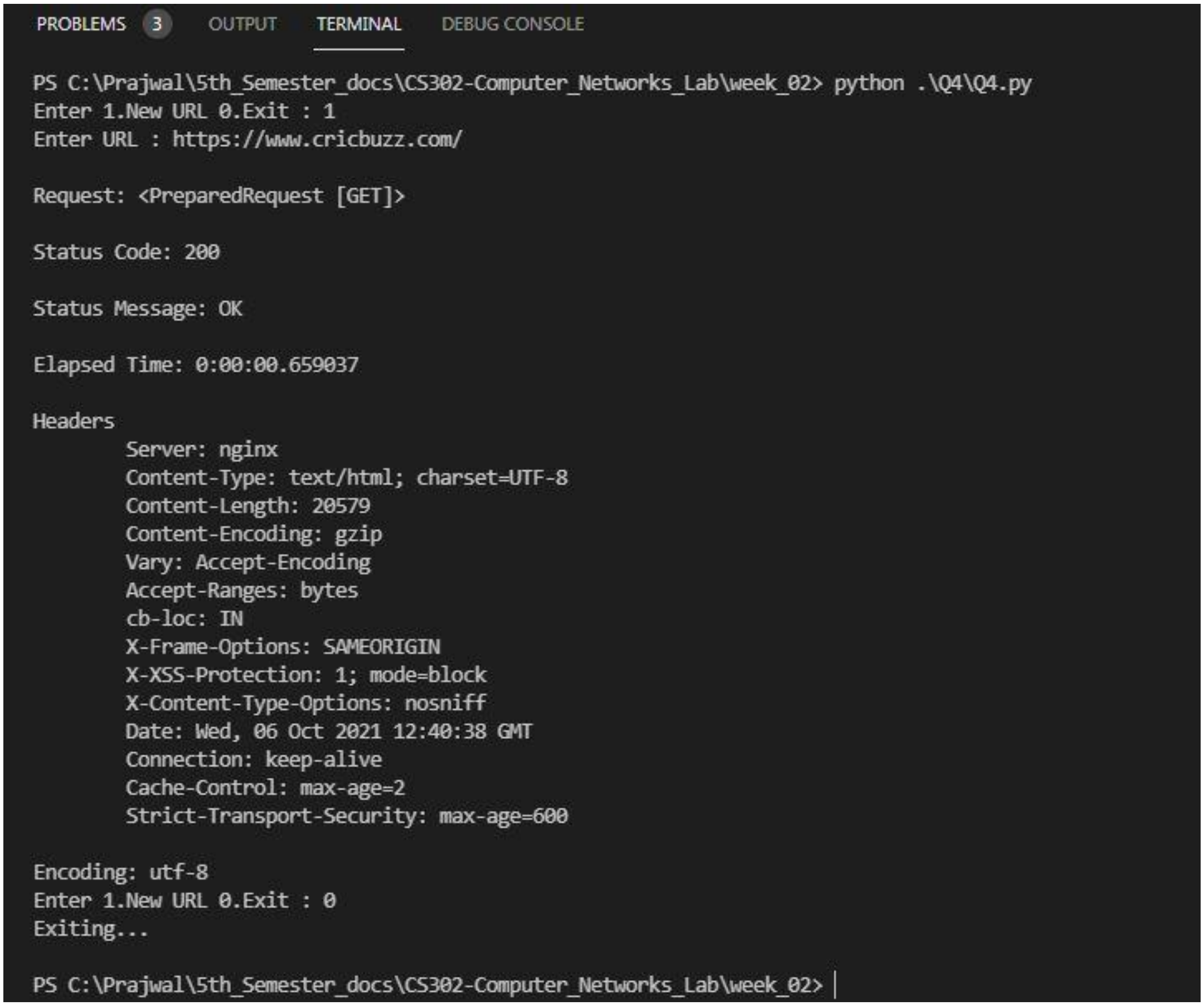
```
        print(f"\nStatus Message: {res.reason}")
        print(f"\nElapsed Time: {elapsedTime}")
        print(f"\nHeaders")
        for a in res.headers:
            print("\t" + a + ": " + res.headers[a])

        print(f"\nEncoding: {res.apparent_encoding}")
    except:
        print("\nUnknown Error Occurred\nTry again")

while True:
    choice = int(input("Enter 1.New URL 0.Exit : "))
    if choice == 1:
        URL=input("Enter URL : ")
        printURLDetails(URL)
    else:
        print("Exiting...\n")
        break
```

b) Explanation:-

Using get function of requests.api we can get all the details of the inputted URL. Then we just extract the details of the URL like request type, status code, HTTP headers etc and display it.

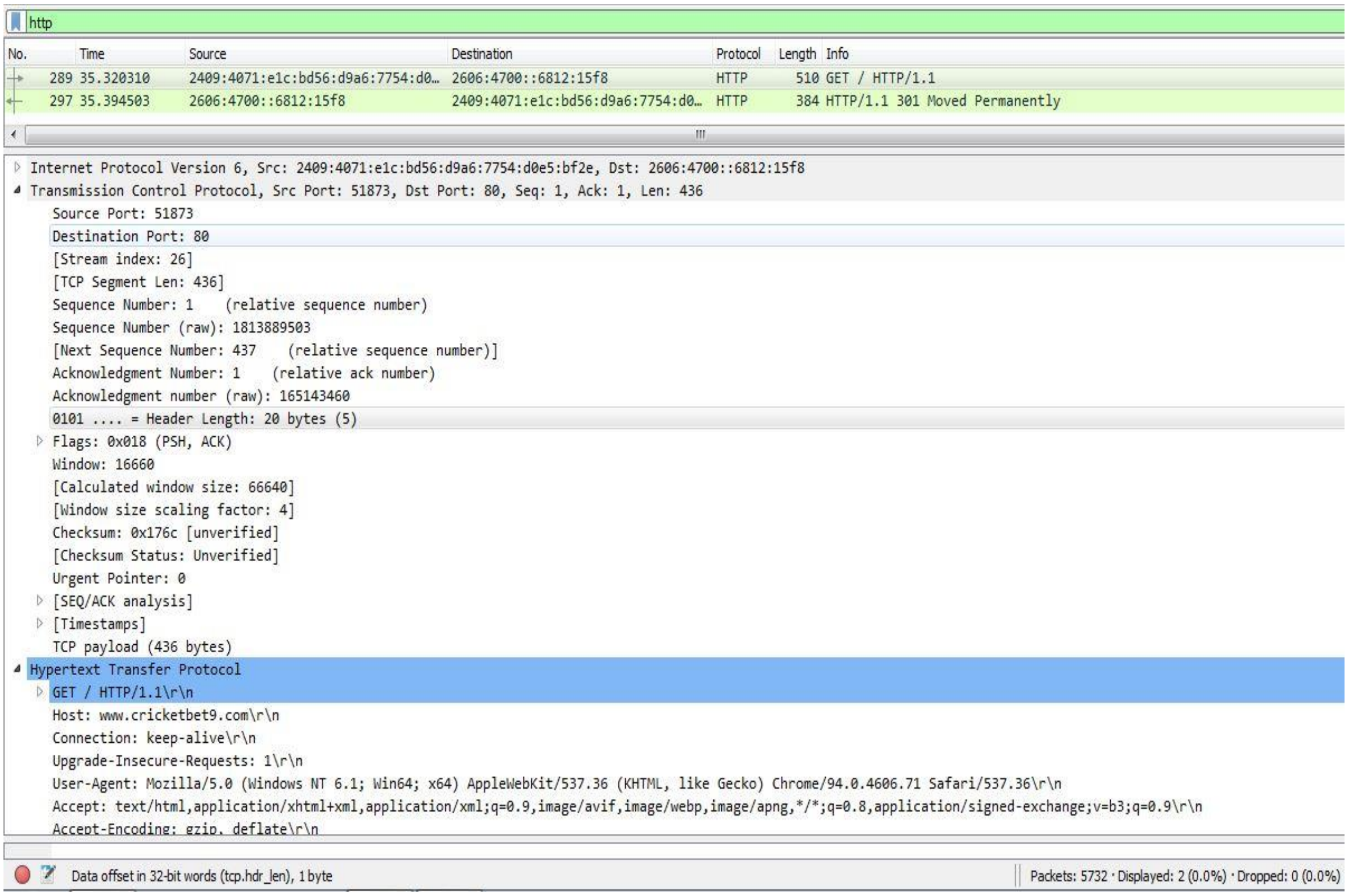
(P.T.O)

c) Output:-



```
PS C:\Prajwal\5th_Semester_docs\CS302-Computer_Networks_Lab\week_02> python .\Q4\Q4.py
Enter 1.New URL 0.Exit : 1
Enter URL : https://www.cricbuzz.com/

Request: <PreparedRequest [GET]>

Status Code: 200

Status Message: OK

Elapsed Time: 0:00:00.659037

Headers
        Server: nginx
        Content-Type: text/html; charset=UTF-8
        Content-Length: 20579
        Content-Encoding: gzip
        Vary: Accept-Encoding
        Accept-Ranges: bytes
        cb-loc: IN
        X-Frame-Options: SAMEORIGIN
        X-XSS-Protection: 1; mode=block
        X-Content-Type-Options: nosniff
        Date: Wed, 06 Oct 2021 12:40:38 GMT
        Connection: keep-alive
        Cache-Control: max-age=2
        Strict-Transport-Security: max-age=600

Encoding: utf-8
Enter 1.New URL 0.Exit : 0
Exiting...

PS C:\Prajwal\5th_Semester_docs\CS302-Computer_Networks_Lab\week_02> |
```

Q5) Capture HTTP packet using wireshark and analyze the packet and significance of its various fields.



```
No.   Time       Source                           Destination                      Protocol Length Info
  289 35.320310  2409:4071:e1c:bd56:d9a6:7754:d0… 2606:4700::6812:15f8             HTTP        510 GET / HTTP/1.1
  297 35.394503  2606:4700::6812:15f8             2409:4071:e1c:bd56:d9a6:7754:d0… HTTP        384 HTTP/1.1 301 Moved Permanently
```

```
▷ Internet Protocol Version 6, Src: 2409:4071:e1c:bd56:d9a6:7754:d0e5:bf2e, Dst: 2606:4700::6812:15f8
◢ Transmission Control Protocol, Src Port: 51873, Dst Port: 80, Seq: 1, Ack: 1, Len: 436
    Source Port: 51873
    Destination Port: 80
    [Stream index: 26]
    [TCP Segment Len: 436]
    Sequence Number: 1    (relative sequence number)
    Sequence Number (raw): 1813889503
    [Next Sequence Number: 437    (relative sequence number)]
    Acknowledgment Number: 1    (relative ack number)
    Acknowledgment number (raw): 165143460
    0101 .... = Header Length: 20 bytes (5)
  ▷ Flags: 0x018 (PSH, ACK)
    Window: 16660
    [Calculated window size: 66640]
    [Window size scaling factor: 4]
    Checksum: 0x176c [unverified]
    [Checksum Status: Unverified]
    Urgent Pointer: 0
  ▷ [SEQ/ACK analysis]
  ▷ [Timestamps]
    TCP payload (436 bytes)
◢ Hypertext Transfer Protocol
  ▷ GET / HTTP/1.1\r\n
    Host: www.cricketbet9.com\r\n
    Connection: keep-alive\r\n
    Upgrade-Insecure-Requests: 1\r\n
    User-Agent: Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/94.0.4606.71 Safari/537.36\r\n
    Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n
    Accept-Encoding: gzip, deflate\r\n
```

```
Data offset in 32-bit words (tcp.hdr_len), 1 byte        Packets: 5732 · Displayed: 2 (0.0%) · Dropped: 0 (0.0%)
```

The above is a HTTP packet captured by wireshark. In this we can see the Transport layer protocol used is Transport Control Protocol (TCP). source port is 51873 and destination port is 80. The Sequence number and Acknowledgement number is also shown which is special property used in TCP.

Then there is HTTP details given below which is the Application layer protocol.

- The HTTP request type is GET

- The path requested is ‘/’
- The version of HTTP protocol used is HTTP 1.1
- The host to which the request is made is ‘www.cricketbet9.com’
- Connection header is keep-alive which signifies that the HTTP connection established is persistent type of HTTP connection.