

# CS 331: Computer Networks

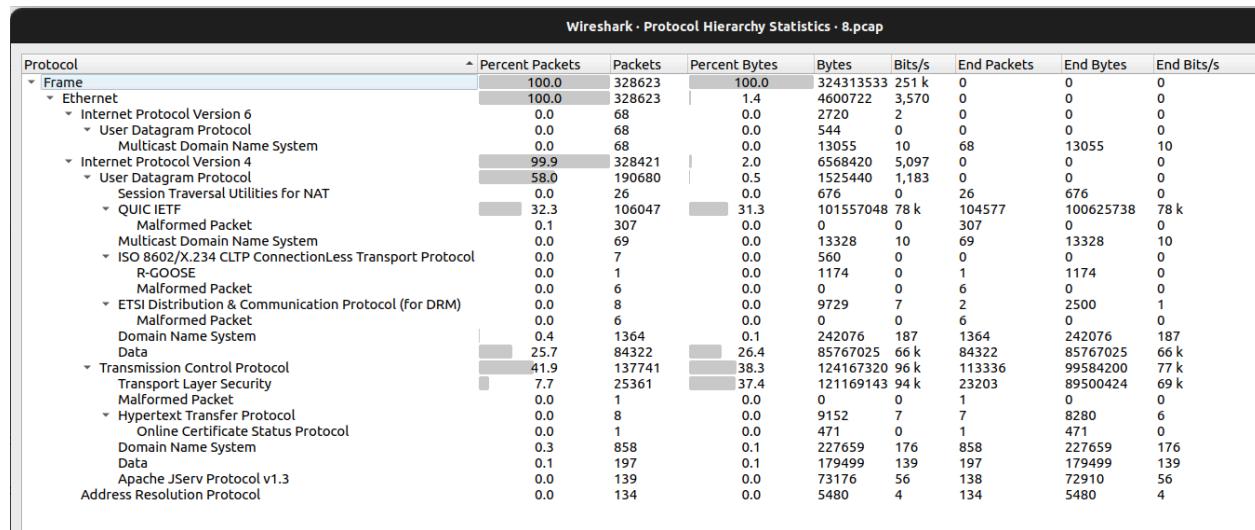
## Assignment 1

Team 17: Hrriday Ruparel (22110099), Kishan Ved (22110122)

[Repository Link](#)

## Part 1: Metrics and Plots

For this question, we made a sniffer using `c++`. Packets were sent via `tcpreplay` on interface `lo`. The sniffer simply stored captured packets in a `pcap` file without processing them. **It was observed that every packet is sniffed twice (ie; every packet is duplicated).** We believe this happens because the same packet is sniffed when sent from and when received by the interface `lo`. Also, our code sniffs TCP and UDP based packets since we observed that all the packets contain only TCP and UDP as Transport Layer Protocols.



Upon interaction with other groups, it was found that using a different interface enables sniffing the packets only once. However, when we tried sniffing on the only other interface available on our laptops (`wlp` interface), the sniffing process was **extremely slow (<100 pps)**. The same

code when run on a laptop with `eht0` interface sniffs fast and only one packet at a time. Thus, we have used `lo`, and then run a simple command to drop all duplicate packets.

```
editcap -d input.pcap output.pcap
```

The `output.pcap` obtained after doing this contains no duplicate packets. The following questions are answered based on the analysis of the `output.pcap` file. In all the codes written for performing analysis on the `output.pcap` file, the packets with source or destination IP as `127.0.0.1` are ignored. These packets are extra packets that are captured while using the `lo` interface. We observed that `8.pcap` has no packets with `127.0.0.1` as the source or the destination ip, hence, ignoring these does not change the answers.

For our team, the `input.pcap` file was `8.pcap`

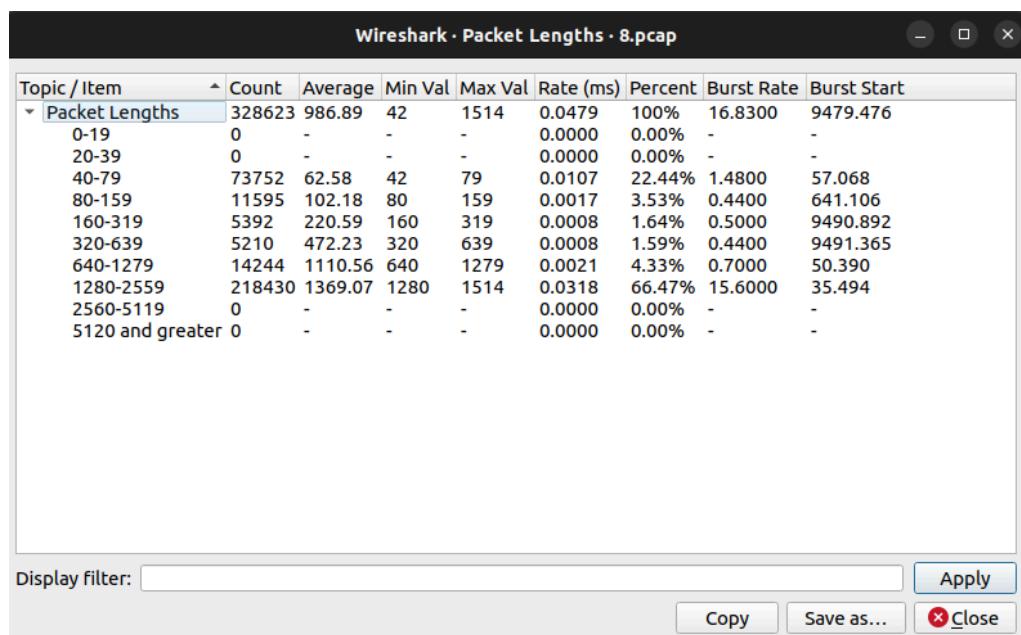
[All media / images used in this report can be found in the GitHub repository](#)

Q1) Find the total amount of data transferred (in bytes), the total number of packets transferred, and the minimum, maximum, and average packet sizes. Also, show the distribution of packet sizes (e.g., by plotting a histogram of packet sizes).

A)

**Total Data Transferred:** 323,446,252 bytes      (Actual Data Size: 329,571,525 bytes)  
**Total Packets Received:** 327,775      (Actual Number of Packets: 328,623)  
**Minimum Packet Size:** 42 bytes      (Actual Min Packet Size: 42 bytes)  
**Maximum Packet Size:** 1514 bytes      (Actual Max Packet Size: 1514 bytes)  
**Average Packet Size:** 986.794 bytes      (Average Packet Size: 986.89 bytes)

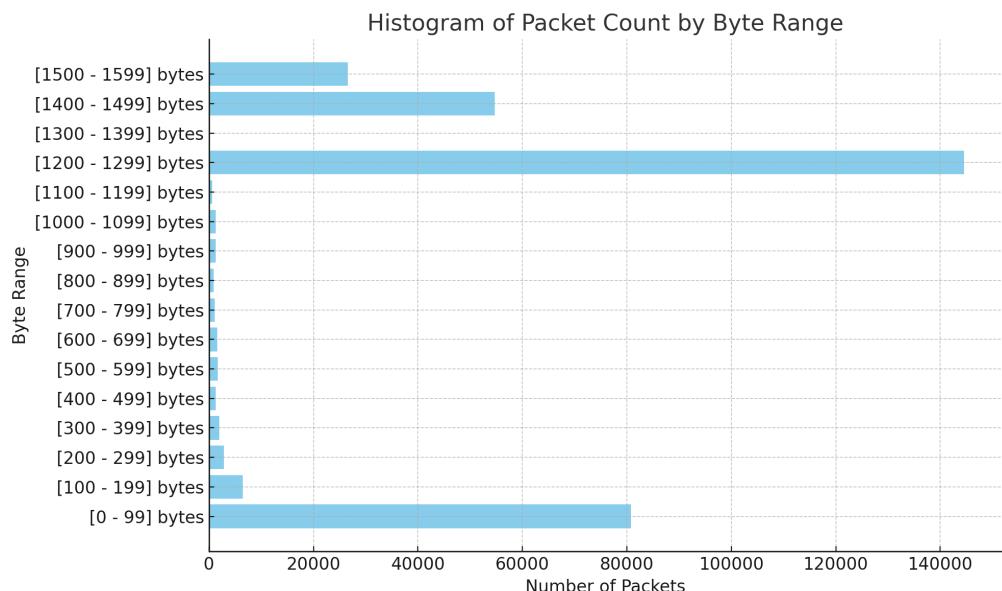
Original `8.pcap` Statistics



Packet Size Distribution (Histogram) from `output.pcap`:

[0 - 99] bytes: 80740 packets  
[100 - 199] bytes: 6477 packets  
[200 - 299] bytes: 2870 packets  
[300 - 399] bytes: 1985 packets  
[400 - 499] bytes: 1296 packets  
[500 - 599] bytes: 1711 packets  
[600 - 699] bytes: 1603 packets  
[700 - 799] bytes: 1048 packets  
[800 - 899] bytes: 871 packets  
[900 - 999] bytes: 1325 packets  
[1000 - 1099] bytes: 1285 packets  
[1100 - 1199] bytes: 637 packets  
[1200 - 1299] bytes: 144545 packets  
[1300 - 1399] bytes: 166 packets  
[1400 - 1499] bytes: 54659 packets  
[1500 - 1599] bytes: 26557 packets

The above data is used to plot this histogram



Q2) Find unique source-destination pairs (source IP:port and destination IP:port) in the captured data.

A) Number of unique source-destination pairs: **4439 pairs**

Q3) Display a dictionary where the key is the IP address and the value is the total flows for that IP address as the source. Similarly display a dictionary where the key is the IP address and the value is the total flows for that IP address as the destination. Find out which source-destination (source IP:port and destination IP:port) have transferred the most data

A)

Dictionaries are uploaded here: [https://github.com/Kishan-Ved/cn\\_a1/blob/main/p1q3\\_flows.txt](https://github.com/Kishan-Ved/cn_a1/blob/main/p1q3_flows.txt)

Flow with the most data transferred:

**Source IP:** 54.230.112.80:443 -> **Destination IP:** 10.7.11.235:61198 transferred **72893512 bytes**

Q4) List the top speed in terms of ‘pps’ and ‘mbps’ that your program is able to capture the content without any loss of data when i) running both tcpreplay and your program on the same machine (VM), and ii) when running on different machines: Two student group should run the program on two different machines eg. tcpreplay on physical-machine of student1 and sniffer program physical-machine of student2. Single students should run between two VMs.

A)

For the same machine with WiFi turned off and interface set to `lo` we were able to achieve:

- **10000 pps (with packet loss of <800 packets)**
- **1000 pps (no packet loss)**

To replicate the scenario between 2 different machines, we have used a **docker container**, and run tcpreplay within the container. The packets were captured by running the sniffer in the host machine, outside the docker container. The interface of the docker container had a different IP compared to the host machine. The WiFi was turned off during the replay and the docker bridge was used to establish connection between the docker container and host machine . Please refer to this [video](#). The video shows the transfer of some packets from the docker container to the local machine.

# Part 2: Catch Me If You Can

Q1) The absolute difference of the Source port and Destination port value is given as 54286, The ACK flag is set, and the last 4 digits of the Acknowledgement number is ‘1203’. Find the Source IP and Destination IP.

A)

Protocol: TCP

Source IP: 10.7.11.235, Destination IP: 180.149.59.137

Source Port: 54729, Destination Port: 443

Protocol: TCP

Source IP: 10.7.11.235, Destination IP: 180.149.59.137

Source Port: 54729, Destination Port: 443

**Total number of packets found: 2**

Q2) The first two digits of the checksum is ‘b5’ ( it is of the form 0xb5\_\_\_\_, hexadecimal notation), the urgent data pointer is set to 0 and the last 4 four digits of Raw Sequence Number is ‘6183’. Find the source IP, destination IP and the complete checksum of this packet.

A)

Protocol: TCP

Source IP: 23.54.155.211, Destination IP: 10.7.11.235

Checksum: 0xb55a, Sequence number: 1322276183

Verification with original 8.pcap

The screenshot shows a Wireshark capture window with the following details:

- File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
- ip.src==23.54.155.211 & ip.dst==10.7.11.235
- No. Time Source Destination Protocol Length Info
- 150064 - 663.885119 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=335314 Ack=2649 Win=26214
- 150065 - 663.885119 23.54.155.211 10.7.11.235 TLSV1.3 1514 Application Data, Application Data
- 150067 - 663.884427 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=338234 Ack=2649 Win=26214
- 150068 - 663.884427 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=330694 Ack=2649 Win=26214
- 150070 - 663.884169 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=341154 Ack=2649 Win=26214
- 150071 - 663.884169 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=342614 Ack=2649 Win=26214
- 150073 - 663.883754 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=344074 Ack=2649 Win=26214
- 150074 - 663.883754 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=345534 Ack=2649 Win=26214
- 150075 - 663.881355 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=346994 Ack=2649 Win=26214
- 150076 - 663.881355 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=348454 Ack=2649 Win=26214
- 150077 - 663.881355 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=349914 Ack=2649 Win=26214
- 150079 - 663.881105 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=349914 Ack=2649 Win=26214
- 150080 - 663.881105 23.54.155.211 10.7.11.235 TCP 1514 443 → 54337 [PSH, ACK] Seq=351374 Ack=2649 Win=26214
- 150081 - 663.881105 23.54.155.211 10.7.11.235 TLSV1.3 1514 Application Data, Application Data

Sequence Number (raw): 1322276183  
[Next Sequence Number: 335694 (relative sequence number)]  
Acknowledgment Number: 2649 (relative ack number)  
Acknowledgment number (raw): 722659488  
Offset .... = Header Length: 20 bytes (5)  
Flags: 0x018 (PSH, ACK)  
000 ..... = Reserved: Not set  
...0 ..... = Nonce: Not set  
...0..... = Congestion Window Reduced (CWR): Not set  
....0.... = ECN-Echo: Not set  
....0.... = Urgent: Not set  
....1.... = Acknowledgment: Set  
....1.... = Push: Set  
....0.... = Reset: Not set  
....0.... = Syn: Not set  
....0.... = Fin: Not set  
[TCP Flags: ....AP....]  
Window: 4096  
[Calculated window size: 262144]  
[Window size scaling factor: 64]  
Checksum: 0xb55a [Unverified]  
[Checksum Status: Unverified]  
Urgent Pointer: 0  
[Timestamps]  
Time since first frame in this TCP stream: 0.395423000 seconds  
0030 10 00 b5 5a 00 00 82 6b 46 1e bd 85 76 21 7d ef ...Z...k F...v1:-  
Details at: https://www.wireshark.org/docs/wsug\_html\_chunked/ChAdvChecksums.html (tcp.checksum), 2 bytes

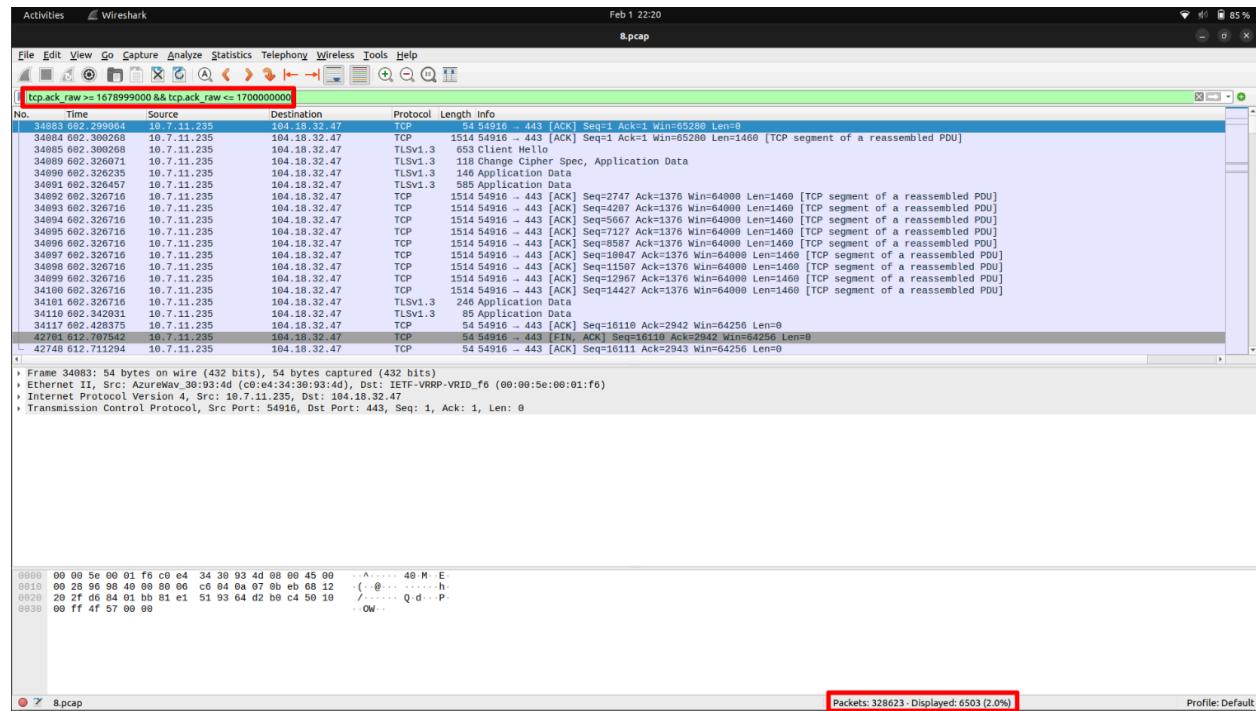
Q3) Tell the number of packets where the sum of source and destination ports is between 10,000 and 20,000.

A) 69 packets

Q4) Find all such packets whose acknowledgement number (raw) is in between 1678999000 <= ack\_number <= 1700000000.

A) 6503 packets

Verification with original 8.pcap



# Part 3: Capture the Packets

Q1) Run the Wireshark tool and capture the trace of the network packets on your host device. We expect you would be connected to the Internet and perform regular network activities.

List at least 5 different application layer protocols that we have not discussed so far in the classroom and describe in 1-2 sentences the operation/usage of protocol and its layer of operation and indicate the associated RFC number if any.

A)

The captured pcap file named `q3.pcap` (captured using Wireshark) is available at:  
[https://github.com/Kishan-Ved/cn\\_a1/blob/main/q3.pcap](https://github.com/Kishan-Ved/cn_a1/blob/main/q3.pcap)

Some application layer protocols not discussed in classroom are:

- **SSH (Secure SHell) [RFC 4251]**
  - SSH is used to securely log into remote machines over an unsecured network, providing encrypted communication. It typically operates on port 22.
  - There are 3 main components to it:
    - The Transport Layer Protocol: SSH-TRANS
    - The User Authentication Protocol: SSH-USERAUTH
    - The Connection Protocol: SSH-CONNECT
  - SSH and its variant protocols work at multiple levels of the network stack.
- **LDAP (Lightweight Directory Access Protocol) [RFC 4511]**
  - LDAP is used for accessing and managing directory services, such as storing user credentials in a directory. It operates primarily over TCP and UDP port 389.
  - It is a lightweight, simplified version of DAP. It is more efficient and easier to implement, especially in environments with limited resources like low-bandwidth networks.
  - Widely used in modern applications, such as for user authentication and directory services.
- **SNMP (Simple Network Management Protocol) [RFC 1157]**
  - SNMP is used for network management, allowing network devices to be monitored and controlled remotely.
  - SNMP is used to communicate management information between the network management stations and the agents in the network elements.
- **AnaSIP (Session Initiation Protocol) [RFC 2543]**
  - SIP is used for signaling and controlling multimedia communication sessions, such as voice and video calls over IP networks.

- Platforms like Zoom and Google Meet use this protocol to set up and manage real-time communication sessions.
- **HLS (HTTP Live Streaming) [RFC 8216]**
  - HLS is a non-standard protocol used for streaming multimedia content over HTTP. It breaks the content into smaller chunks, allowing adaptive bitrate streaming to improve the user experience.
  - Used by live streaming platforms like Twitch, YouTube and Meta and OTT platforms like Apple TV+ and Netflix for streaming VOD.

Q2) Analyze the following details by visiting the following websites in your favourite browser:

- i) canarabak.in
- ii) github.com
- iii) netflix.com

- a. Identify 'request line' with the version of the application layer protocol and IP address. Also, identify whether the connection(s) is/are persistent or not.

A)

### Canara Bank

The screenshot shows the Canara Bank homepage with a large blue overlay asking for language preference between Hindi and English. The browser's developer tools Network tab is open, showing various requests and responses. A specific request to the homepage is highlighted with a red box, showing the following details in the Headers section:

```

Request URL: http://canarabank.com/
Request Method: GET
Status Code: 200 OK
Remote Address: 107.162.160.84:43
Referrer Policy: strict-origin-when-cross-origin
  
```

The Response Headers section also has a red box around it, showing:

```

HTTP/1.1 200 OK
Cache-Control: private, max-age=3600
Content-Type: text/html; charset=UTF-8
  
```

Request line: GET / HTTP/1.1

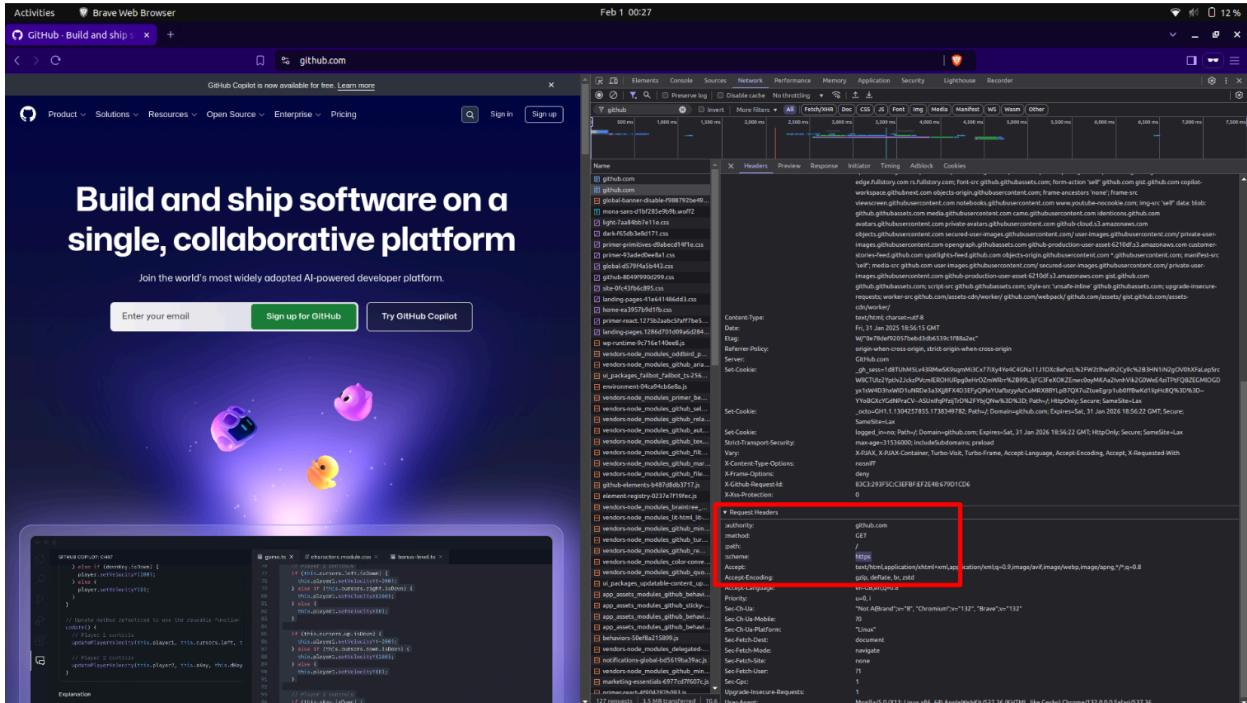
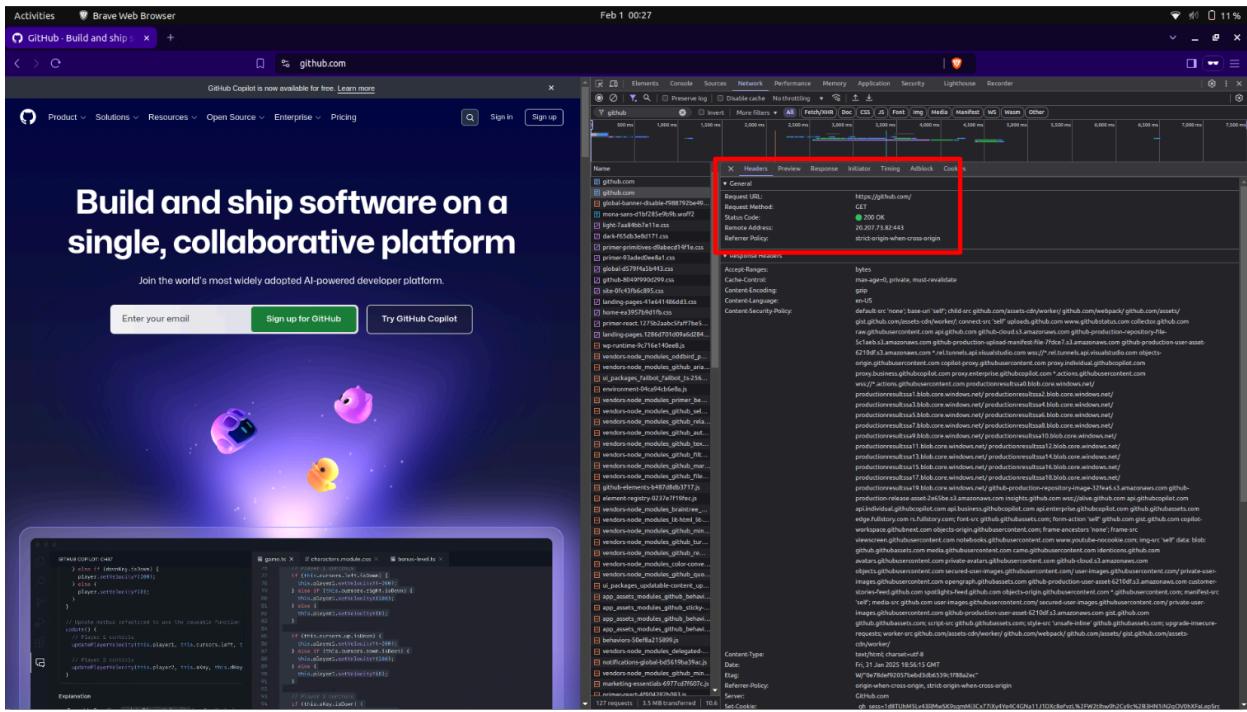
Version: HTTP/1.1

IP Address: 107.162.160.8

Port: 443

Connection: Persistent (keep-alive)

## Github



## Request line: GET / HTTPS

Version: HTTPS

IP Address: 20.205.243.166

Port: 443

#### Connection: Persistent (keep-alive)

Netflix

The screenshot shows the Netflix India homepage. At the top, there's a banner with various movie and TV show thumbnails. Below the banner, a large purple box with white text reads "Unlimited movies, TV shows and more". Underneath that, it says "Starts at ₹149. Cancel at any time." A red button labeled "Get Started >" is visible. To the left of the button is a text input field for an "Email address". Above the main content, the browser's developer tools are open, showing the Network tab with several requests listed. One request for "www.netflix.com/in/" is highlighted with a red box, showing details like the URL, method (GET), status code (200 OK), and remote address (18.200.8.190:443). The Response Headers section also contains a red box highlighting the "Content-Type" header set to "text/html; charset=UTF-8".

The screenshot shows the Netflix India homepage with a banner for 'Friends With Benefits' and 'Mission Impossible'. A large central call-to-action says 'Unlimited movies, TV shows and more'. Below it, it says 'Starts at ₹149. Cancel at any time.' and 'Ready to watch? Enter your email to create or restart your membership.' An 'Email address' input field and a 'Get Started >' button are present. To the right, the browser's developer tools Network tab is open, showing a request to 'https://www.netflix.com/lni'. The request headers include 'authority: www.netflix.com', 'method: GET', 'path: /lni', and 'scheme: https'. The response time is listed as 6,000 ms.

## Request line: GET /in/ HTTPS

Version: HTTPS

IP Address: 54.74.73.31

Port: 443

#### Connection: Persistent (keep-alive)

- b. For any one of the websites, list any three header field names and corresponding values in the request and response message. Any three HTTP error codes obtained while loading one of the pages with a brief description

The screenshot shows a browser window with the GitHub Copilot landing page. The Network tab of the developer tools is open, showing a request to `https://github.com/`. The Headers section is highlighted with a red box, displaying the following fields:

- `authority: github.com`
- `method: GET`
- `path: /`
- `Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8`

### Request Header Fields:

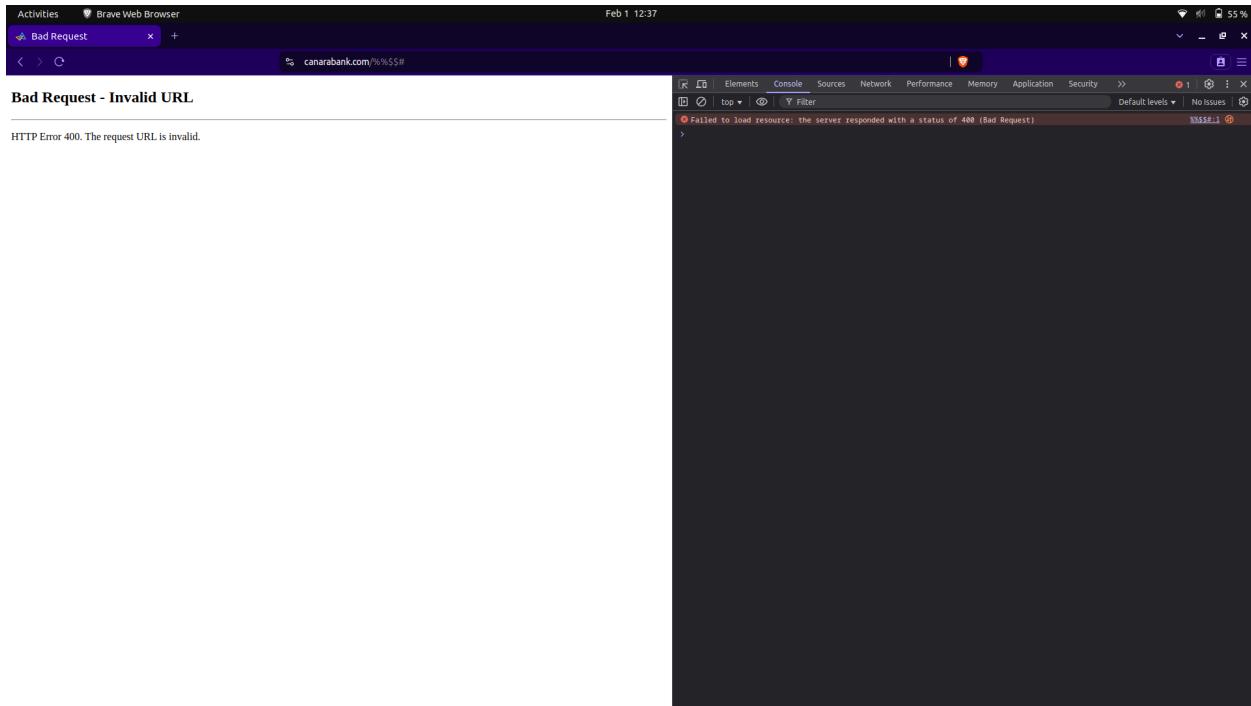
1. `authority: github.com`
2. `method: GET`
3. `path: /`

## Response Header Fields:

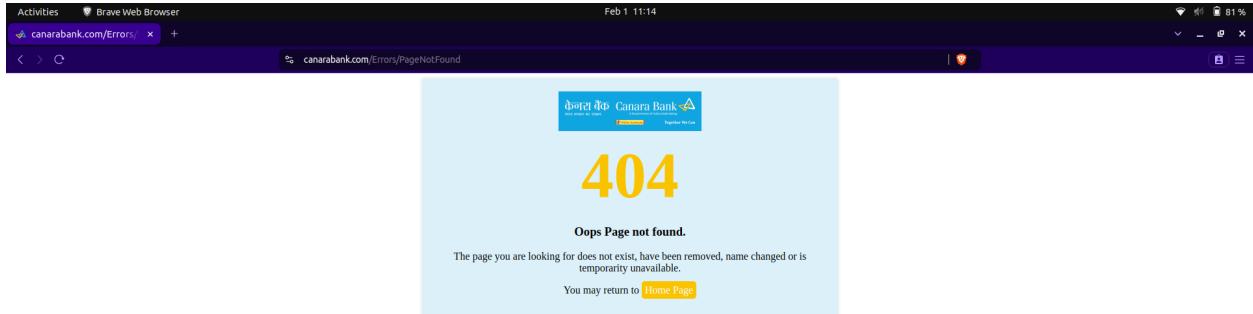
1. Content-type: text/html; charset =utf-8
2. Server: GitHub.com
3. Date: Fri, 31 Jan 2025 18:56:15 GMT

Three HTTP Error codes encountered are:

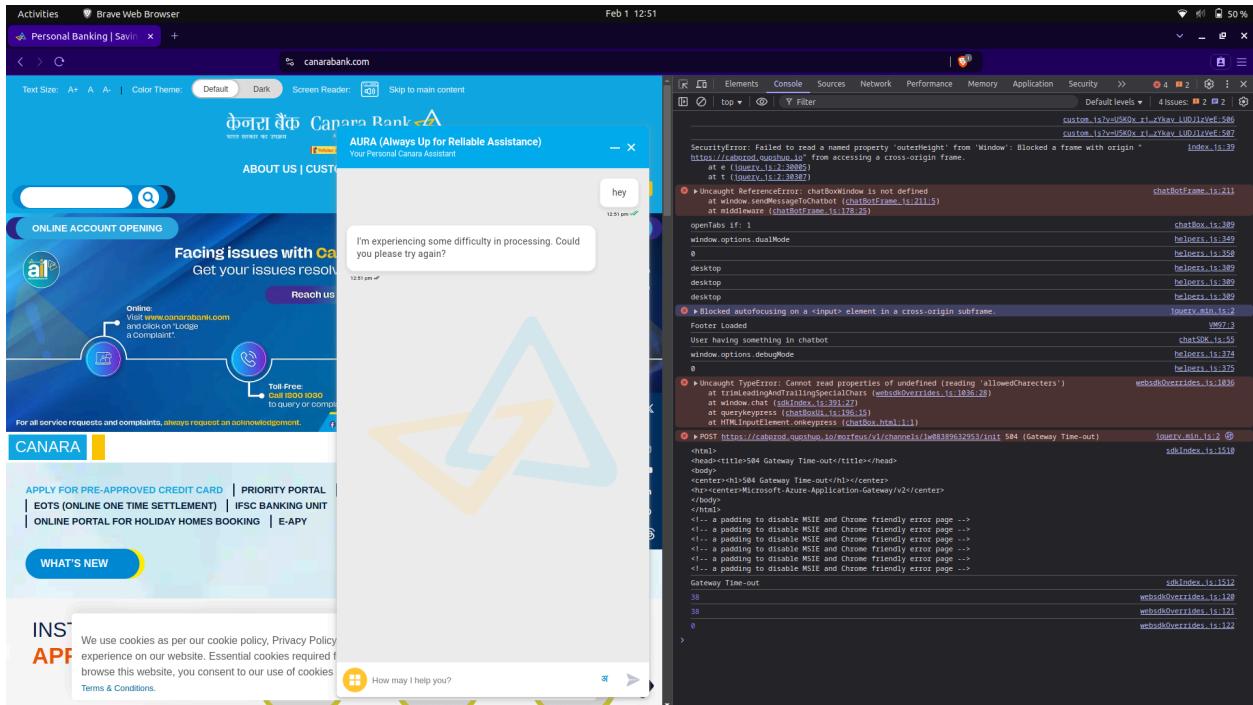
1. **Error 400: Bad Request:** This is a client-side HTTP error and the reason for a 400 response is typically due to malformed request syntax, invalid request message framing, or deceptive request routing.



2. **Error 404: Page Not Found:** This is another client-side HTTP error response code indicating that the server cannot find the requested resource. Links that lead to a 404 page are often called broken or dead links.



3. **Error 504: Gateway Timeout:** This server error response code indicates that the server, while acting as a gateway or proxy, did not get a response in time from the upstream server in order to complete the request.



- c. Capture the Performance metrics that your browser records when a page is loaded and also report the list the cookies used and the associated flags in the request and response headers. Please report the browser name and screenshot of the performance metrics reported for any one of the page loads.

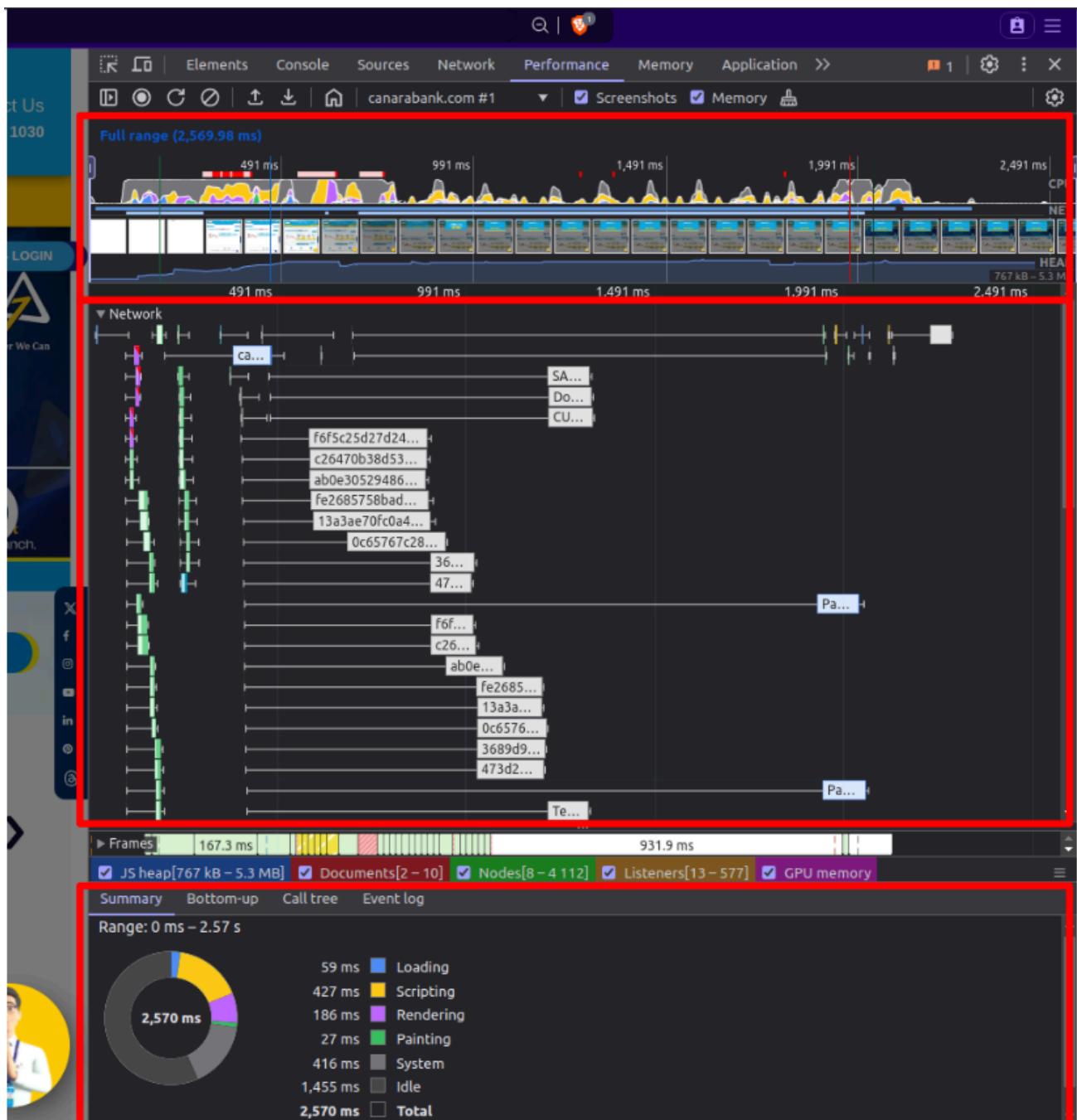
**Browser used:** Brave  
**Cookies used for github.com**

Name	Value	Domain	Path	Expires	Size	Https	Secure	SameSite	Partially Signed	Cross-Site	Prior...
__Host-user_session_same_site	Dm0jTfVuus_wvOIL168M23...	github.com	/	2025...	77	✓	✓	Strict	✓	✓	Medi...
_device_id	a5008fb0eaad944afefaff1...	github.com	/	2025...	42	✓	✓	Lax	✓	✓	Medi...
_oh_sess	UXArhQSNyCpvz%7Tdn...	github.com	/	Sessi...	470	✓	✓	Lax	✓	✓	Medi...
_ecto	GH.1.1.100516602/173748...	github.com	/	2025...	32	✓	✓	Lax	✓	✓	Medi...
color_mode	#7B%2C00%20,color_mode%21%	github.com	/	Sessi...	214	✓	✓	Lax	✓	✓	Medi...
cpu_bucket	md	github.com	/	Sessi...	12	✓	✓	Lax	✓	✓	Medi...
dotcom_user	Robohrriday	github.com	/	2025...	22	✓	✓	Lax	✓	✓	Medi...
logged_in	yes	github.com	/	2025...	12	✓	✓	Lax	✓	✓	Medi...
preferred_color_mode	dark	github.com	/	Sessi...	24	✓	✓	Lax	✓	✓	Medi...
saved_user_sessions	85882839J3ADM8fTVu0...	github.com	/	2025...	78	✓	✓	Lax	✓	✓	Medi...
tz	Asia%2FCalcutta	github.com	/	Sessi...	17	✓	✓	Lax	✓	✓	Medi...
user_session	Dm0jTfVuus_wvOIL168M23...	github.com	/	2025...	60	✓	✓	Lax	✓	✓	Medi...

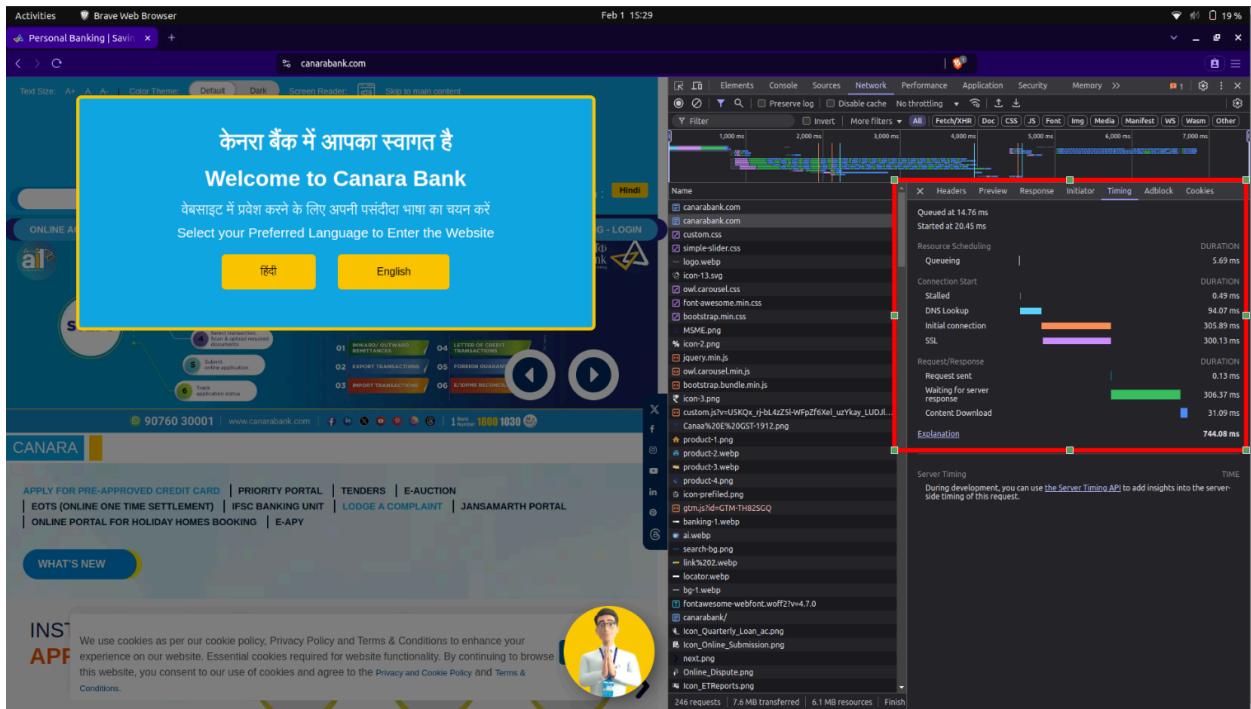
## Request and Response Headers of HTTP request to GET main page of canarabank.com

Name	Value
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding	gzip, deflate, br
Accept-Language	en-US,en;q=0.9
Connection	keep-alive
Host	canarabank.com
Sec-Cloud-Mobile	70
Sec-Cloud-Platform	Linux
Sec-Fetch-Dest	document
Sec-Fetch-Mode	navigate
Sec-Fetch-Site	none
Sec-Fetch-User	?
Upgrade-Insecure-Requests	1
User-Agent	Mozilla/5.0 (Windows NT 10.0; AppleWebKit/537.36 (KHTML, like Gecko) Chrome/132.0.6562.104 Safari/537.36)

## Network performance and summary while loading canarabank.com (entire profile)



## Network performance for a single http request while loading canarabank.com



## Acknowledgements

Some codes are inspired by GitHub repositories, StackOverflow and ChatGPT.