

Programming Assignment 2 : WWW - Report

This assignment contains 4 parts such as “simple web client”, “simple web proxy server”, “simple web server” and the enhancement in the simple proxy server “extended web proxy (Choice-3)”.

PART 1 - Simple web client

- Firstly the web client establishes the TCP connection with the web server directly or through an intermediate web proxy server according to the arguments given to the client. The client sends HTTP requests and uses non-persistent HTTP if the response contains multiple objects.
- After getting the html index file, by using “BeautifulSoup” library we are parsing the html index file and creating the list of links (http) if any. Then for each link from the list we are establishing the TCP connection and closing (non-persistent HTTP). Finally printing the response.
- The arguments given to web client are as follows:
IP address of proxy
IP address of server:Port of server
Port of proxy
Path
- If we don't want to use a proxy then simply give information of the server instead of the proxy and keep the info of the server as it is.

PART 2 - Simple web proxy server

- When the web proxy server is in utilisation there are at least 2 TCP connections (client - proxy and proxy - server). The TCP connection may increase depending on the no. of object files.
- Here the web proxy server is configured to handle multiple clients hence threading is used. Now the proxy decodes the ip and port for the server to which it establishes the TCP connection. Then sends the requests which were provided by the client.
- As soon as the web server returns the object to the proxy, proxy immediately forwards it back to the client
The web proxy server handles response code such as 200 (OK), 404 (Not Found) and 304 (Not Modified). The server responds with the 404 and 304 html file if those cases are arised.

PART 3 - Simple web server

- As soon as the client's (may be client or proxy) request is accepted. If there is a requirement for the separate TCP connections (multiple objects in the response file) then by threading the responses are handled and then sent accordingly. If the server doesn't have the file it simply responds with the 404 response code.

PART 4 - Extension to Simple web proxy server (Choice - 3)

A. URL and Content Filtering at web proxy

- In the beginning of the code we have created a list of the blocked websites and a dictionary for the keywords that we want to shade out with the letter "X".
- The functioning of the extended web proxy server is the same as a simple web proxy server but before requesting the web server it checks if it is in the list of blocked websites or not. If yes then simply respond with the "blocked_website.html" file to the client or else get the response from the server as done in simple web proxy server.
- After receiving the object from the web server the extended web proxy server finds the keyword which needs to be shade out with the letter "X". Then after this modification the response is sent to the client.

B. Web usage stats at web proxy

- The extended proxy server keeps track of browsing activities of users like which websites they have visited, date when they have visited and the no. of time the website is visited and those are stored in the json file as the below given format. Identifying the users with their IP addresses.

```
{
  "127.0.0.1":
  {
    "accessed_websites": [
      {
        "website": "www.example.com",
        "access_date": "2023-11-01"
      },.....],
    "website_access_count":
    {
      "www.example.com": 2,
      "www.iith.ac.in": 1,
      "www.cricinfo.com": 1,
      "www.cse.iith.ac.in": 1
    }
  },
  "192.168.0.1": {....
}
```

Similarly for different users...

- We have created a separate python script called, plot.py to handle the stats display feature. In this we load the json file created by the extended proxy which has the data of the web traffic passing through the proxy. Then we process the data and generate various data points. These data-points (like total number of sites visited by each user, total sites visited by a single user, total sites visited in a date range etc) are then plotted using matplotlib.

Names of files created for each Part: -

For Part 1: -

Script - HTTP_Client.py

We are making a GET request to send in this script. We can send the request directly to server or send to via proxy as well

For Part 2: -

Script - Proxy_Server.py - Simple proxy to take request and forward it to servers and vice-versa for responses

For Part 3: -

Script - HTTP_Server.py

HTML files - hello.html - this is the html file which is present on the '/hello.html' path and will be sent back when asked for it in the GET request.

This file has references to 2 other HTML files well - 'ref_file_1.html' , 'ref_file_2.html'

For Part 4: -

Script - HTTP_Proxy_ext.py - Extended proxy with 'Website blocking', 'word filtering' and web statistics functionality.

Blacklisted_response.html - this is the html file which is sent back to client of it sends a request to a website which is in the block_website list in the proxy.

User_statistics.json - This is the json file which is created by extended proxy and has web-traffic related data.

plot.py - This is the script which takes the json file, processes the web-traffic data and plots different graphs accordingly

For Parts 1, 2, and 3 (Simple web client, server and proxy): -

Client sending request directly to website: -

[illegible]

Client sending request directly to localhost web-server: -

```

kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x v
kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Project$ python3 HTTP_Client.py 127.0.0.1 127.0.0.1:8080 8080 /hello.html
>'HTTP/1.1 200 OK\r\nContent-Length: 281\r\n\r\n<DOCTYPE html>\n\n<html lang="en">\n\n<head>\n\n<meta charset="UTF-8" />\n\n<title>Hello, world!</title>\n\n</head>\n\n<body>\n\n<h1>Hello, world!</h1>\n\n<div>\n\n<a href="/ref_file_1.html">ref File 1</a>\n\n</div>\n\n<div>\n\n<div>\n\n</div>\n\n</body>\n\n</html>'\n\nSending request to: http://127.0.0.1:8080/ref_file_1.html
>'HTTP/1.1 200 OK\r\nContent-Length: 160\r\n\r\n<DOCTYPE html>\n\n<html lang="en">\n\n<head>\n\n<meta charset="UTF-8">\n\n<title>reference File 1</title>\n\n</head>\n\n<body>\n\n<h1>Ref File 1 body</h1>\n\n</body>\n\n</html>'\n\nSending request to: http://127.0.0.1:8080/ref_file_2.html
>'HTTP/1.1 200 OK\r\nContent-Length: 160\r\n\r\n<DOCTYPE html>\n\n<html lang="en">\n\n<head>\n\n<meta charset="UTF-8">\n\n<title>Reference File 2</title>\n\n</head>\n\n<body>\n\n<h1>Ref File 2 body</h1>\n\n</body>\n\n</html>'\n\nkaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Project$

```

Note since the hello.html file also had references to 2 other html files, client sent non-persistent request to fetch those files as well and have received the responses

Server side logs showing 200 OK status codes after send the response: -

```
Kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Prj... × kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Prj... × kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Prj... ×
kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Prj...$ python3 HTTP_Server.py
Web server is running on port 8080...
file to send - hello.html
content read from file - b'<!DOCTYPE html>\n<html lang="en">\n<head>\n  <meta charset="UTF-8" />\n  <title>Hello, world!</title>\n</head>\n<body>\n  <h1>Hello, world!</h1>\n</body>\n</html>\n'
response to send - HTTP/1.1 200 OK
Content-Length: 281

b'<!DOCTYPE html>\n<html lang="en">\n<head>\n  <meta charset="UTF-8" />\n  <title>Hello, world!</title>\n</head>\n<body>\n  <h1>Hello, world!</h1>\n</body>\n</html>\n'
file to send - ref_file_1.html
content read from file - b'<!DOCTYPE html>\n<html lang="en">\n<head>\n  <meta charset="UTF-8">\n  <title>Reference File 1</title>\n</head>\n<body>\n  <h1>Ref File 1 body</h1>\n</body>\n</html>\n'
response to send - HTTP/1.1 200 OK
Content-Length: 160

b'<!DOCTYPE html>\n<html lang="en">\n<head>\n  <meta charset="UTF-8">\n  <title>Reference File 1</title>\n</head>\n<body>\n  <h1>Ref File 1 body</h1>\n</body>\n</html>\n'
file to send - ref_file_2.html
content read from file - b'<!DOCTYPE html>\n<html lang="en">\n<head>\n  <meta charset="UTF-8">\n  <title>Reference File 2</title>\n</head>\n<body>\n  <h1>Ref file 2 body</h1>\n</body>\n</html>\n'
response to send - HTTP/1.1 200 OK
Content-Length: 160

b'<!DOCTYPE html>\n<html lang="en">\n<head>\n  <meta charset="UTF-8">\n  <title>Reference File 2</title>\n</head>\n<body>\n  <h1>Ref file 2 body</h1>\n</body>\n</html>\n'

```

Client sending request to localhost web-server via proxy: -

Client side request and response logs: -

```
kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x
kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Project$ python3 HTTP_Client.py 127.0.0.1 127.0.0.1:8080 8888 /hello.html
b'HTTP/1.1 200 OK\r\nContent-Length: 281\r\n\r\n<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>Hello, world!</title>\r\n</head>\r\n  <body>\r\n    <h1>Hello, world!</h1>\r\n    <div>\r\n      <a href="/ref_file_1.html">ref File 1</a>\r\n    </div>\r\n    <div>\r\n      <a href="/ref_file_2.html">ref File 2</a>\r\n    </div>\r\n  </body>\r\n</html>\r\n'
Sending request to: http://127.0.0.1:8080/ref_file_1.html
b'HTTP/1.1 200 OK\r\nContent-Length: 160\r\n\r\n<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>reference File 1</title>\r\n</head>\r\n  <body>\r\n    <h1>Ref File 1 body</h1>\r\n  </body>\r\n</html>\r\n'
Sending request to: http://127.0.0.1:8080/ref_file_2.html
b'HTTP/1.1 200 OK\r\nContent-Length: 160\r\n\r\n<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>Reference File 2</title>\r\n</head>\r\n  <body>\r\n    <h1>Ref file 2 body</h1>\r\n  </body>\r\n</html>\r\n'
kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Project$
```

Proxy logs showing the requests received from client and corresponding responses received from server: -

```
kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x
kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Project$ python3 Proxy_Server.py
Proxy server is running on port 8888...
GET /hello.html HTTP/1.0
Host: 127.0.0.1:8080

Parsing request...
command: GET
host: 127.0.0.1 | port: 8080
b'HTTP/1.1 200 OK\r\nContent-Length: 281\r\n\r\n<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>Hello, world!</title>\r\n</head>\r\n  <body>\r\n    <h1>Hello, world!</h1>\r\n    <div>\r\n      <a href="/ref_file_1.html">ref File 1</a>\r\n    </div>\r\n    <div>\r\n      <a href="/ref_file_2.html">ref File 2</a>\r\n    </div>\r\n  </body>\r\n</html>\r\n'
socket closed
GET /ref_file_1.html HTTP/1.0
Host: 127.0.0.1:8080

Parsing request...
command: GET
host: 127.0.0.1 | port: 8080
b'HTTP/1.1 200 OK\r\nContent-Length: 160\r\n\r\n<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>reference File 1</title>\r\n</head>\r\n  <body>\r\n    <h1>Ref File 1 body</h1>\r\n  </body>\r\n</html>\r\n'
socket closed
GET /ref_file_2.html HTTP/1.0
Host: 127.0.0.1:8080

Parsing request...
command: GET
host: 127.0.0.1 | port: 8080
b'HTTP/1.1 200 OK\r\nContent-Length: 160\r\n\r\n<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>Reference File 2</title>\r\n</head>\r\n  <body>\r\n    <h1>Ref file 2 body</h1>\r\n  </body>\r\n</html>\r\n'
socket closed
```

Server side logs showing request received from proxy and 200 OK status codes after sending responses back to proxy: -

```
b'<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>Reference File 2</title>\r\n</head>\r\n  <body>\r\n    <h1>Ref file 2 body</h1>\r\n  </body>\r\n</html>\r\n'
file to send - hello.html
content read from file - b'<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>Hello, world!</title>\r\n</head>\r\n  <body>\r\n    <h1>Hello, world!</h1>\r\n    <div>\r\n      <a href="/ref_file_1.html">ref File 1</a>\r\n    </div>\r\n    <div>\r\n      <a href="/ref_file_2.html">ref File 2</a>\r\n    </div>\r\n  </body>\r\n</html>\r\n'
response to send - HTTP/1.1 200 OK
Content-Length: 281

b'<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>Hello, world!</title>\r\n</head>\r\n  <body>\r\n    <h1>Hello, world!</h1>\r\n    <div>\r\n      <a href="/ref_file_1.html">ref File 1</a>\r\n    </div>\r\n    <div>\r\n      <a href="/ref_file_2.html">ref File 2</a>\r\n    </div>\r\n  </body>\r\n</html>\r\n'
file to send - ref_file_1.html
content read from file - b'<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>reference File 1</title>\r\n</head>\r\n  <body>\r\n    <h1>Ref File 1 body</h1>\r\n  </body>\r\n</html>\r\n'
response to send - HTTP/1.1 200 OK
Content-Length: 160

b'<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>reference File 1</title>\r\n</head>\r\n  <body>\r\n    <h1>Ref File 1 body</h1>\r\n  </body>\r\n</html>\r\n'
file to send - ref_file_2.html
content read from file - b'<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>Reference File 2</title>\r\n</head>\r\n  <body>\r\n    <h1>Ref file 2 body</h1>\r\n  </body>\r\n</html>\r\n'
response to send - HTTP/1.1 200 OK
Content-Length: 160

b'<!DOCTYPE html>\r\n<html lang="en">\r\n<head>\r\n  <meta charset="UTF-8" />\r\n  <title>Reference File 2</title>\r\n</head>\r\n  <body>\r\n    <h1>Ref file 2 body</h1>\r\n  </body>\r\n</html>\r\n'

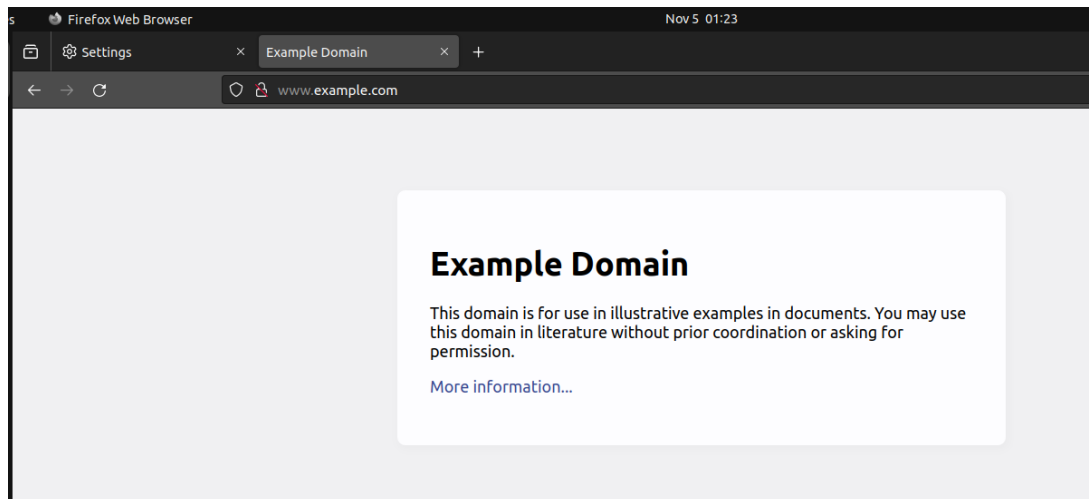
```

Browser sending request to website via proxy: -

Log of proxy showing the browser request and server response of requesting '<http://www.example.com>': -

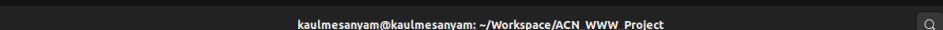
[illegible]

Browser snapshot: -



Browser sending request directly to localhost web-server: -

Server logs: -



The screenshot shows a terminal window with the following content:

```

Nov 5 01:24
Terminal
kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Project

kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Pr... x
kaulmesanyam@kaulmesanyam: ~/Workspace/ACN_WWW_Project$ python3 HTTP_Server.py
Web server is running on port 8080...
file to send - hello.html
content read from file - b'<DOCTYPE html>\n<html lang="en">\n<head>\n  <meta charset="UTF-8" />\n  <title>Hello, world!\n</title>\n</head>\n<body>\n  <h1>Hello, world!\n</h1>\n  <div>\n    <a href="/ref_file_1.html">ref File 1</a>\n  </div>\n  <div>\n    <a href="/ref_file_2.html">ref File 2</a>\n  </div>\n</body>\n</html>'
response to send - HTTP/1.1 200 OK
Content-Length: 281

b'<DOCTYPE html>\n<html lang="en">\n<head>\n  <meta charset="UTF-8" />\n  <title>Hello, world!\n</title>\n</head>\n<body>\n  <h1>Hello, world!\n</h1>\n  <div>\n    <a href="/ref_file_1.html">ref File 1</a>\n  </div>\n  <div>\n    <a href="/ref_file_2.html">ref File 2</a>\n  </div>\n</body>\n</html>'

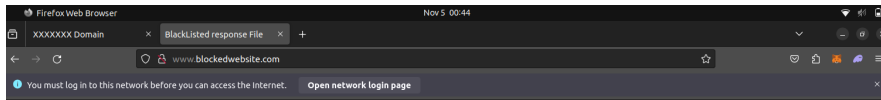
```

Browser snapshot: -



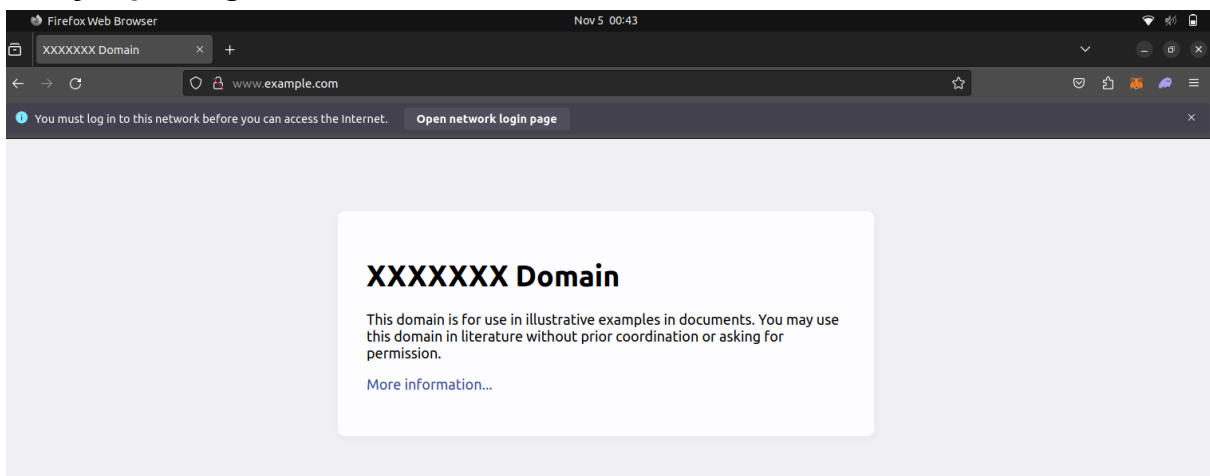
For Part- 4 (Extended proxy) : -

Proxy blocking request to blacklisted website: -

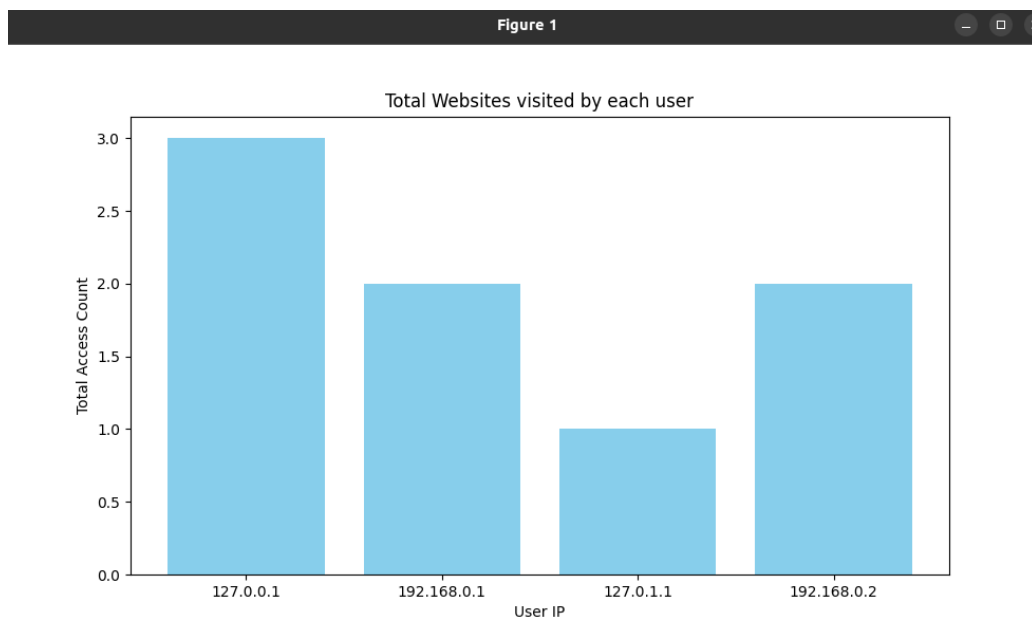


The website you are trying to access is blacklisted!

Proxy replacing censored word with 'XXXX' : -

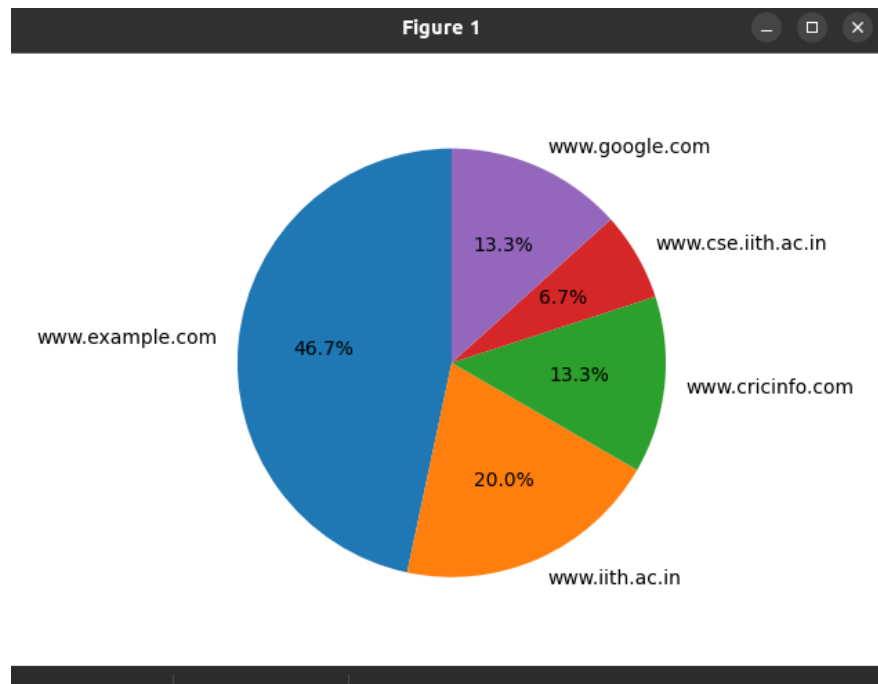


Fetching, processing and plotting of web-traffic data: -

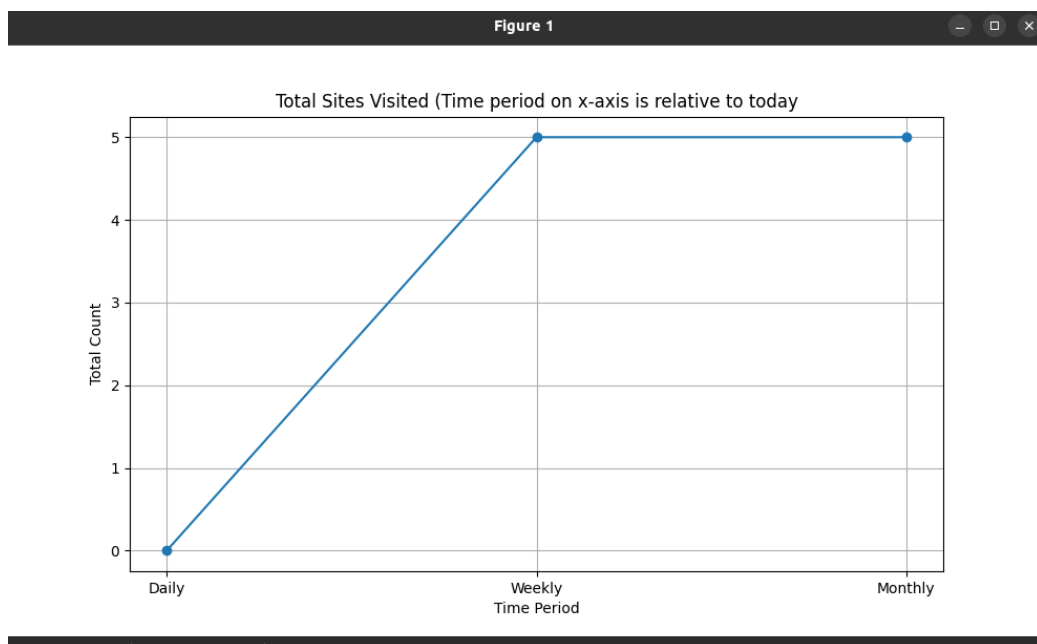


Plot of user vs total website visits

Pie chart for % of websites visited from total web traffic at proxy



Plot for daily / weekly / monthly statistics vs total visits among all the users.



References

We have used LLM (ChatGPT) in order to resolve bugs and implement Python code. Some prompts and responses are as follows: -

i want to store statistics in a json file and fetch it from there

```
# Load existing user statistics from a JSON file
try:
    with open('user_statistics.json', 'r') as file:
        user_statistics = json.load(file)
except FileNotFoundError:
    user_statistics = {}

def save_statistics():
    # Save user statistics to a JSON file
    with open('user_statistics.json', 'w') as file:
        json.dump(user_statistics, file)

def update_statistics(client_address, accessed_website):
    if client_address not in user_statistics:
        user_statistics[client_address] = {'accessed_websites': set(), 'website_access_count':
defaultdict(int)}

    user_statistics[client_address]['accessed_websites'].add(accessed_website)
    user_statistics[client_address]['website_access_count'][accessed_website] += 1
```

**line 355, in raw_decode raise JSONDecodeError("Expecting value", s, err.value) from None
json.decoder.JSONDecodeError: Expecting value: line 1 column 1 (char 0)**

```
{
  "client_address1": {
    "accessed_websites": ["website1", "website2"],
    "website_access_count": {
      "website1": 3,
      "website2": 1
    }
  },
  "client_address2": {
    "accessed_websites": ["website3"],
    "website_access_count": {
      "website3": 5
    }
  }
}
```

line 376, in _iterencode_dict raise TypeError(f'keys must be str, int, float, bool or None, ' TypeError: keys must be str, int, float, bool or None, not tuple

```

import socket
import threading
from collections import defaultdict
import json
import datetime

PORT = 7777
BLACKLIST = {'www.blockedwebsite.com'}
CENSORED = {'Example'}

# Load existing user statistics from a JSON file
try:
    with open('user_statistics.json', 'r') as file:
        user_statistics = json.load(file)
except (FileNotFoundError, json.decoder.JSONDecodeError):
    user_statistics = {}

def save_statistics():
    # Save user statistics to a JSON file
    with open('user_statistics.json', 'w') as file:
        json.dump(user_statistics, file)

def update_statistics(client_address, accessed_website):
    client_address_str = str(client_address)
    if client_address_str not in user_statistics:
        user_statistics[client_address_str] = {'accessed_websites': set(), 'website_access_count': defaultdict(int)}

    user_statistics[client_address_str]['accessed_websites'].add(accessed_website)
    user_statistics[client_address_str]['website_access_count'][accessed_website] += 1

# ... (rest of the code remains the same)

```

give me a sample website_counts this function will return

```

start_date = "2022-01-01"
end_date = "2022-01-31"
data = {
    1: {"accessed_websites": [{"website": "example.com", "access_date": "2022-01-05"},
                              {"website": "example.com", "access_date": "2022-01-15"},
                              {"website": "example.net", "access_date": "2022-01-20"}]},
    2: {"accessed_websites": [{"website": "example.com", "access_date": "2022-01-10"},
                              {"website": "example.org", "access_date": "2022-01-25"}]}
}

website_counts = get_website_counts_in_range(start_date, end_date, data)

```

ANTI-PLAGIARISM Statement

We certify that this assignment/report is our own work, based on our personal study and/or research and that we have acknowledged all material and sources used in its preparation, whether they be books, articles, packages, datasets, reports, lecture notes, and any other kind of document, electronic or personal communication. We also certify that this assignment/report has not previously been submitted for assessment/project in any other course lab, except where specific permission has been granted from all course instructors involved, or at any other time in this course, and that we have not copied in part or whole or otherwise plagiarised the work of other students and/or persons. Additionally, we acknowledge that we may have used AI tools, such as language models (e.g., ChatGPT, Bard), for assistance in generating and refining my assignment, and we have made all reasonable efforts to ensure that such usage complies with the academic integrity policies set for the course. I pledge to uphold the principles of honesty and responsibility at CSE@IITH. In addition, we understand our responsibility to report honour violations by other students if we become aware of it.

Sanyam Kaul - CS23MTECH14011
Bhargav Patel - CS23MTECH11026
Arnab Ghosh - CS23MTECH11025

Date: 5/11/2023

Signatures: