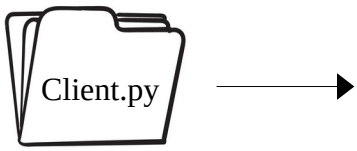
# ACN Programming Assignment-2: WWW

## PART-1: A Simple Web Client




This Python script allows the end user to communicate with the desired destination server and fetch the required resources and associated resources, and provides options to view or save the content as specified by the user, dumped at the server's end. The Local Client is desigend to make just GET requets that to with just HTTP 1.0. It can be useful for web scraping, resource downloading, or analyzing web pages.

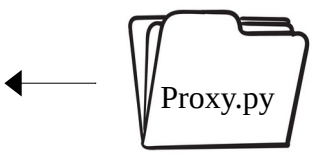
The client takes the following inputs:

1) Server IP/ Domain (Mandatory)
2) Server Port (Mandatory)
3) Proxy IP (optional)
4) Proxy Port (optional)

It is assumed that if the user specifies Proxy entries then it must want to go to destination via proxy server. Based on the input of the user the program generates HTTP/1.0 GET request packet and sends to the destined server. While receiving the corresponding response it first check the status of the response message to verify that whether its requested was served or not by the server. If serverd it seperates the content from the headers and parse the response if it contains html kind of file otherwise it stores what it receives. While parsing if it finds links to the other resources, it in very similar fashion as mentioned above make non-persistent connection with the destined server to fetch the required stuff. At the end the web client stores whatever it recieved in a valid extension with an option to open that in decoded or raw version in browser or command line interface respectively.

## PART-2: A Simple Web Proxy Server

This multithreaded python program acts as a simple proxy server i.e. which just performs Proxying (Forwarding the request) and not caching the content. In other words it just an intermediary thing that takes clients requests and passes it to the destined location and vice versa. When a client request arrives at the proxy server it first seperates the headers and the data content. Then it analyze the headers via some split operation on the headers to detect what is the dedstination host, its port and the path.

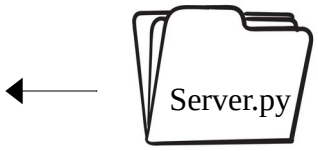



Now, using the extracted values of destiantion host, port and path it will modify the header as the path of the request line that arrived at the proxy is the complete URL and not just path. So this modification of header will convert the URL to path in the request line. After it modifies the HTTP

request it sends it to the destination server and waits for the response in loop. Once it start recieving the response it immediately sends them to the requesting client until no more response is recieved.

## PART-3: A Simple Web Server

This multithreaded Python script is listening for the client's request on a particular port. It is designed to server only GET request so if any user request other then the GET arrives at the local web server the it won't be appreciated by the server. Whenever it recieves a GET request it parse that and checks that whether it has resources mentioned in that request. If so that it formulates the response packet with the required HTTP headers feilds and throws to the requesting client. If not then it sends the status code corresponding to resourse not found along with status phrase.




It is importand to note that there will be a separate TCP connection in a separate thread for each HTTP request/response pair.

## PART-4: Extensions to Simple Proxy Server or web client/web server

**What's our Extension?**

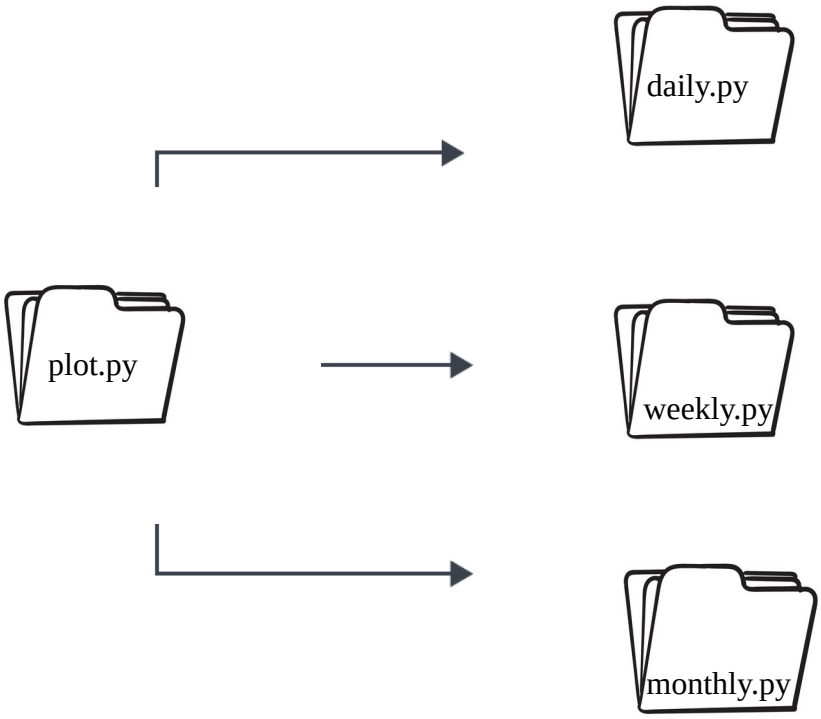
English to Hindi Web PageTranslation and web usage stats at web proxy

**English to Hindi Web PageTranslation**


The Extended proxy server is a simple web proxy server with more enhanced feature of translating the web page's content from english to hindi i.e. if any request that goes through this server then the english text in its response must be translated to hindi and then should be passed on to the client. The only difference here is that first we should accumulate the data instead of sending it directly when we have a packet as transalting the data from one language to another impacts the content legth. So whenever a request arrives similar to the simple proxy server first we modify the request and directs it to the destination. But now we will club all the packets from the destination if the response contains text/html then we must convert the text included in that file to hindi. So we parse all the tag that contain text additionally taking care of tagless text. After transaltion we again modify the response header to have the length of the modified data. Its an important step otherwise client may recieve incomplete content or may indefinitely buffer.

It should be noted that only the content that contains text should be translated and other types of content must be directly forwarded to the client without any translation. Here also client connections are handled concurrently using threads, allowing multiple clients to be served simultaneously.

**Web usage stats at web proxy**




The code runs three external Python scripts to generate statistics, combines the resulting images vertically into a single chart, and offers the user the choice to open and view the combined chart in a web browser or exit the program.

**daily.py plots:**

- **Website with the Highest Count on Each Day (Bar Chart):** This is a bar chart that displays the daily counts for websites. It shows the highest count for each day, and the bars are annotated with the corresponding website names.
- **Daily Count (Line Chart):** The second plot is a line chart showing the daily total counts. It represents the cumulative count of visits for each day.

**weekly.py plots:**

- **Stacked Bar Chart of Visits by Server IPs:** The first plot is a stacked bar chart that displays weekly statistics of website visits categorized by different server IPs. Each bar represents a week, and the heights of the bars are segmented to show the contribution of each server IP to the total count for that week.
- **Box Plot of Visits per Week:** The second plot is a box plot that visualizes the distribution of visit counts per week. It provides insights into the spread and central tendencies of the weekly data.

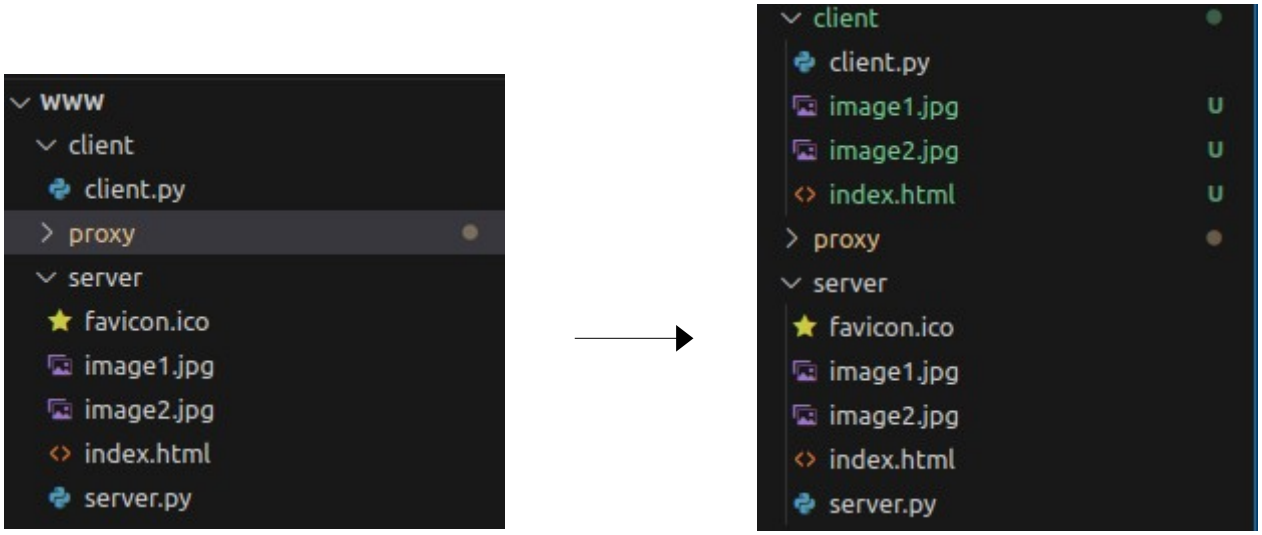
**monthly.py plots**

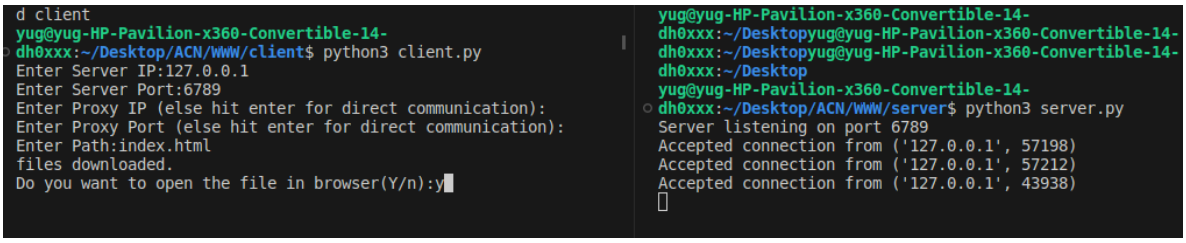
- **Pie Chart -** Number of Users per Month: This is a pie chart that illustrates the distribution of the number of unique users visiting a website for each month. Each slice of the pie represents a month, and the percentage of users for each month is labeled on the chart. The

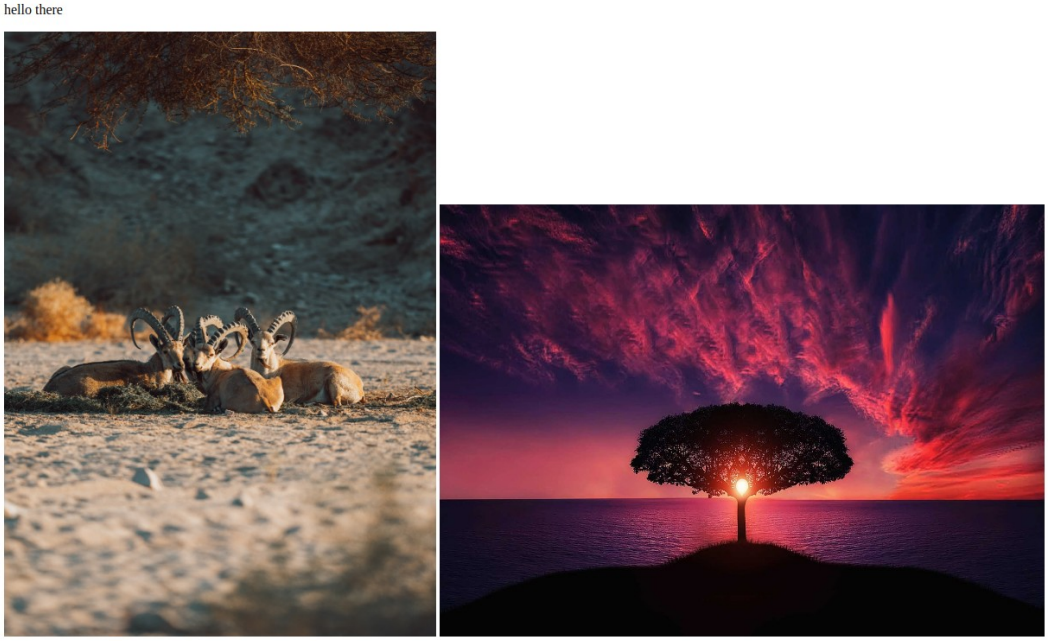
pie chart provides a visual summary of the distribution of users across different months. In summary, the code generates a pie chart that presents the distribution of unique users per month, highlighting the proportion of users for each month in the dataset.

## Some Possible Client – Server Interaction

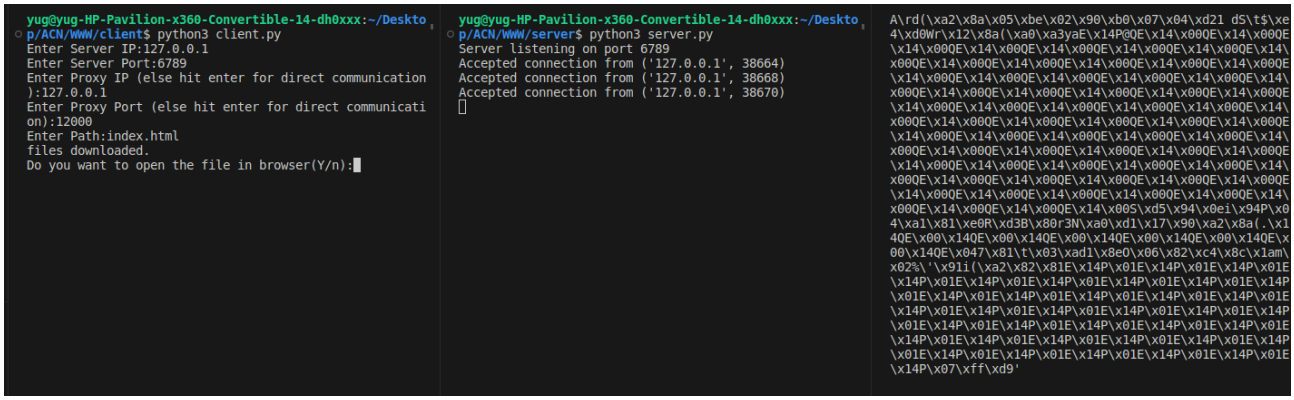
### (1) Local client communicates with local web server directly

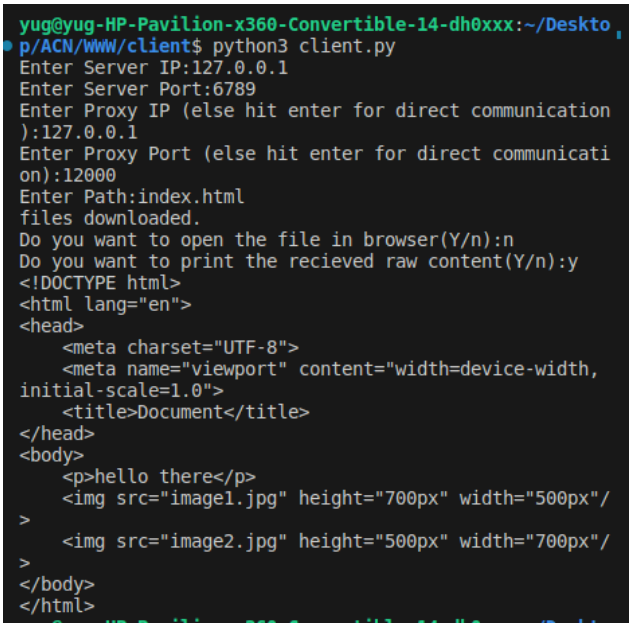


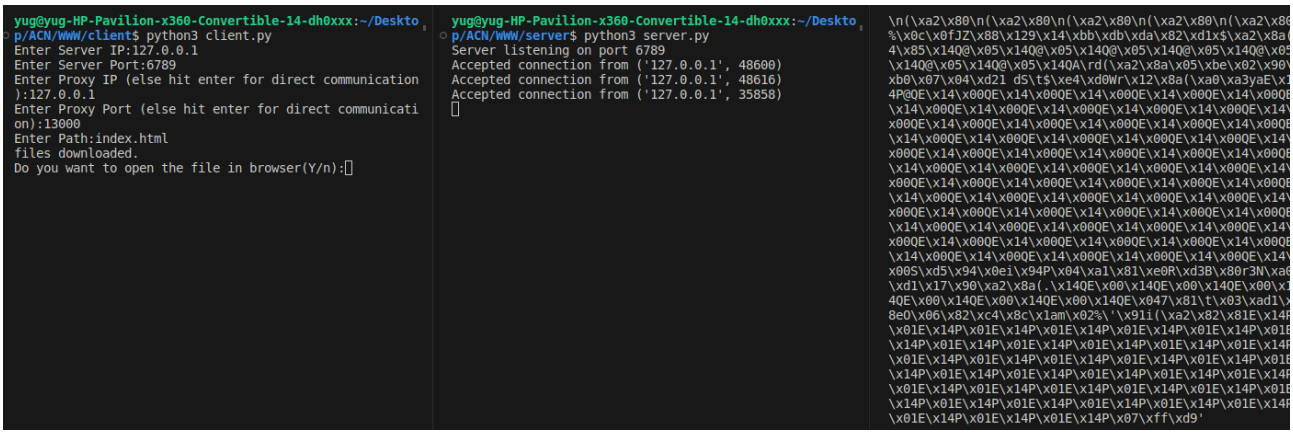



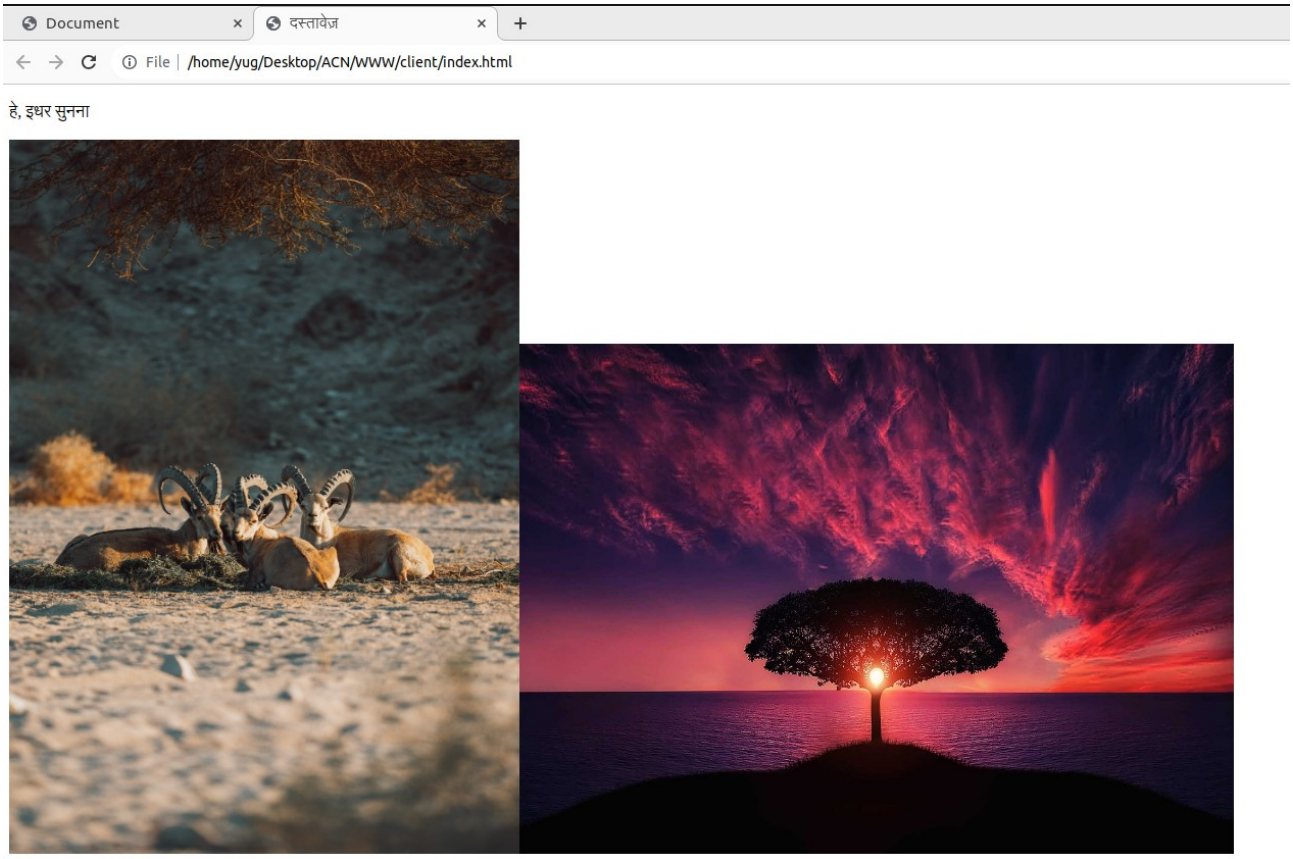
## (2) Local client communicates with local web server via simple proxy



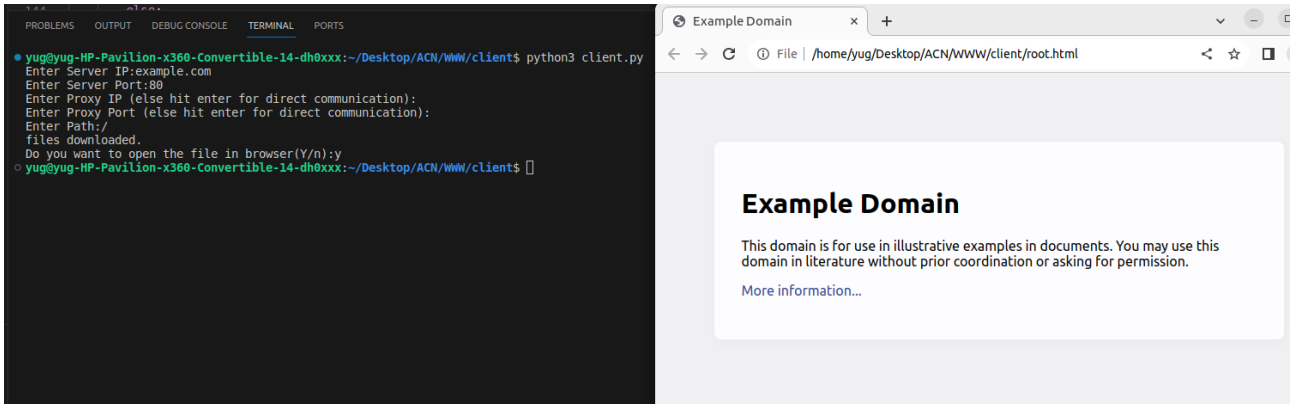




## (3) Local client communicates with local web server via extended proxy

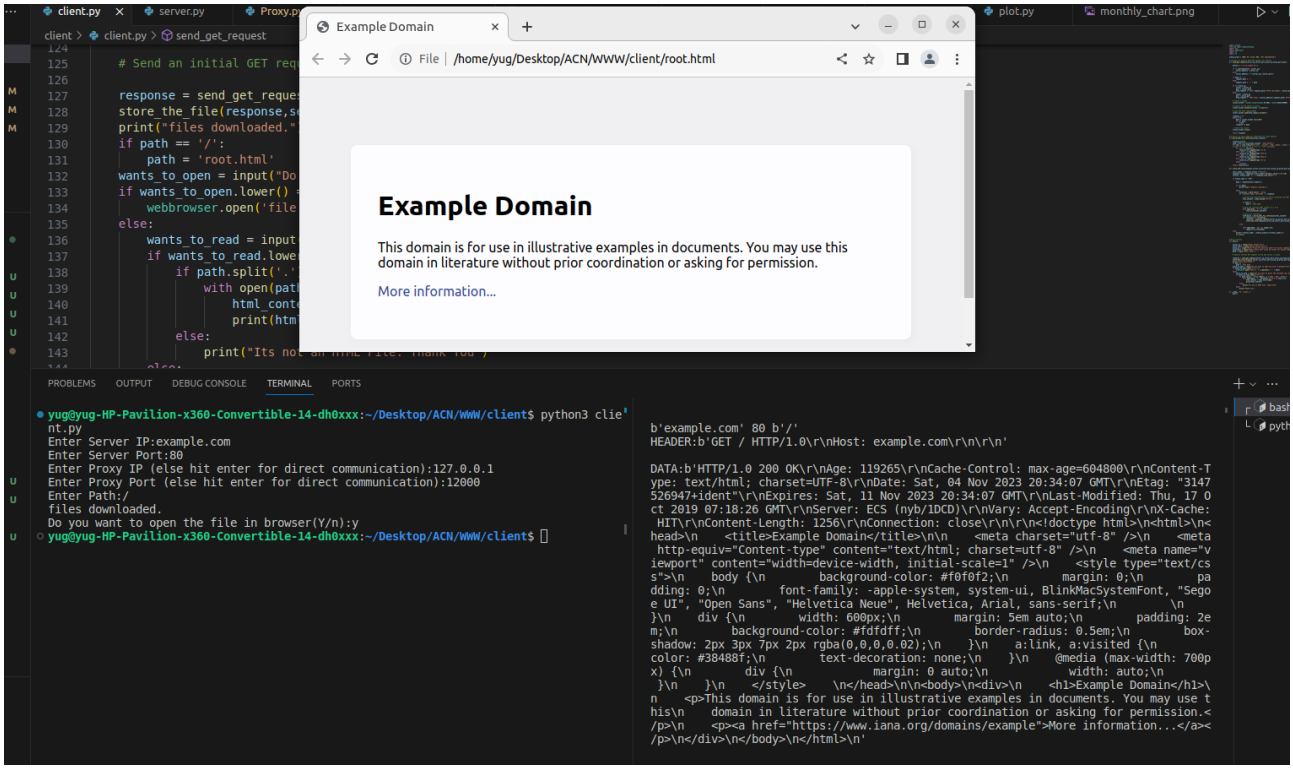




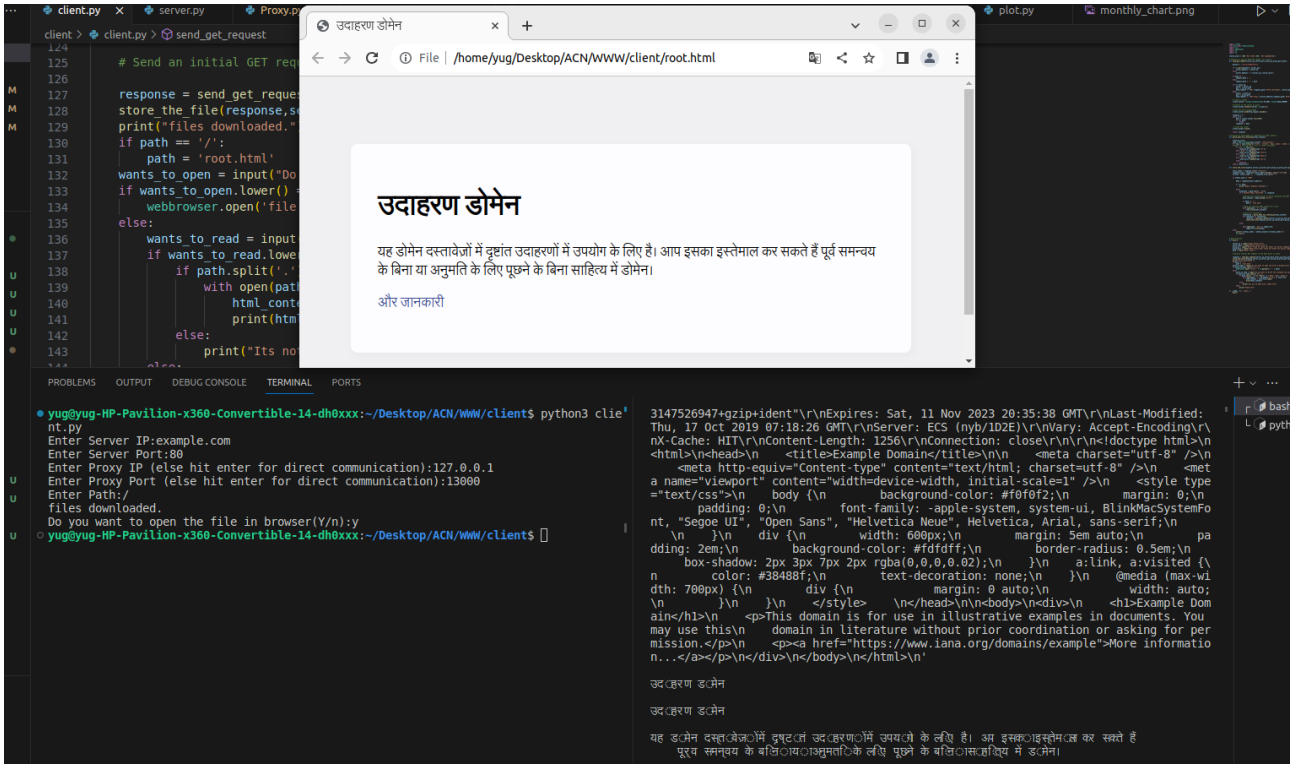

## (4) Local client communicates with remote host (example.com) directly

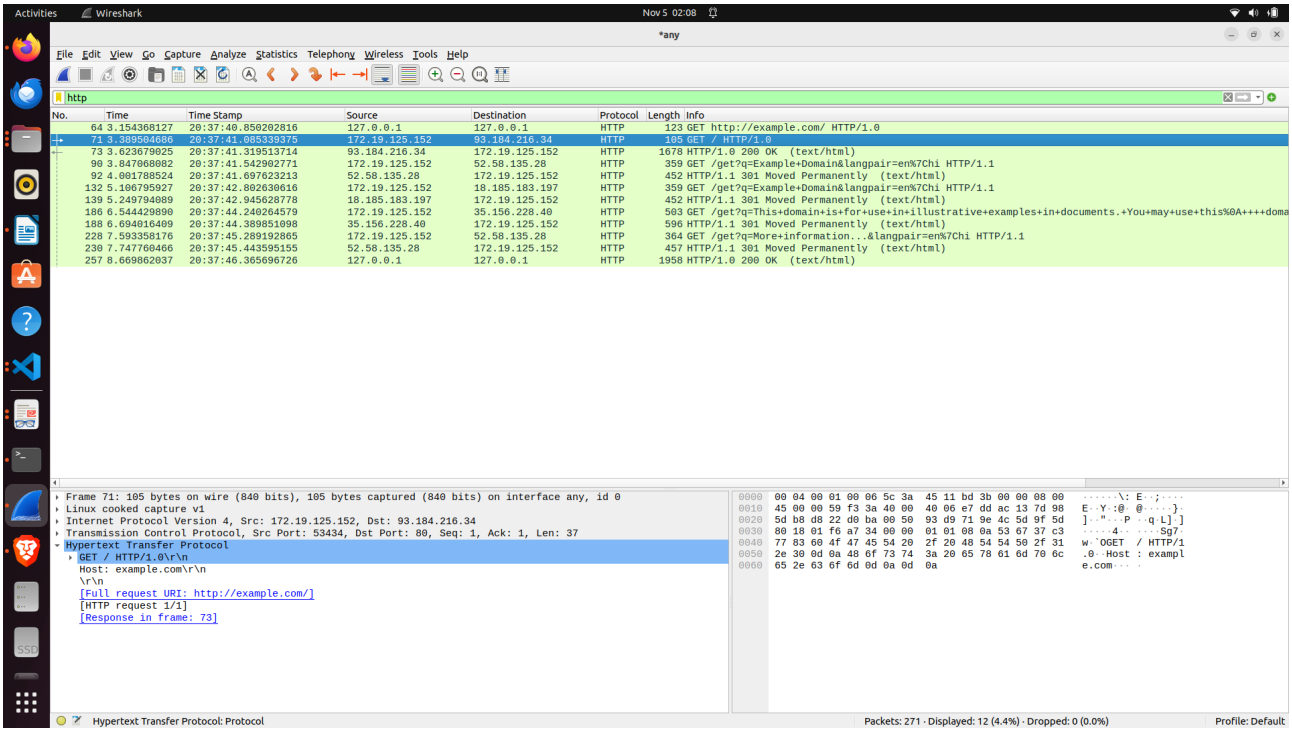


## (5) Local client communicates with remote host (example.com) via simple proxy



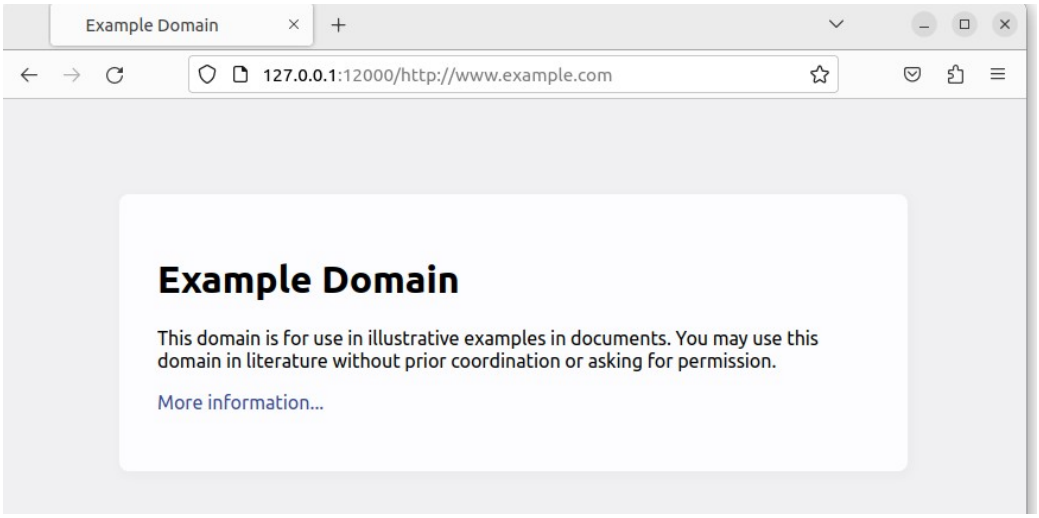

## (6) Local client communicates with remote host (example.com) via Extended proxy





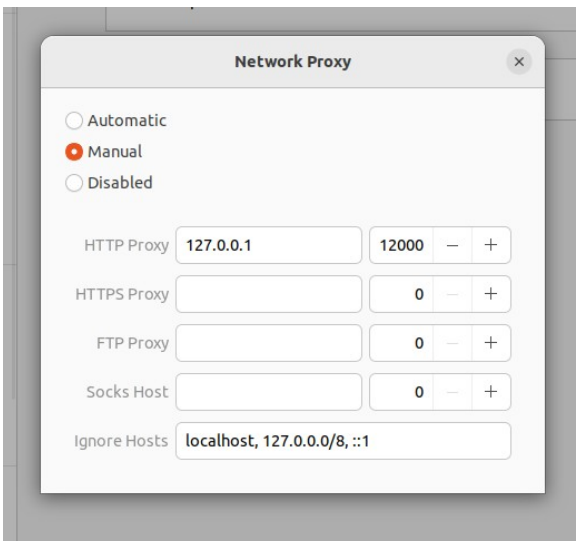

## (7) Browser communicates with local server via simple proxy (explicit)

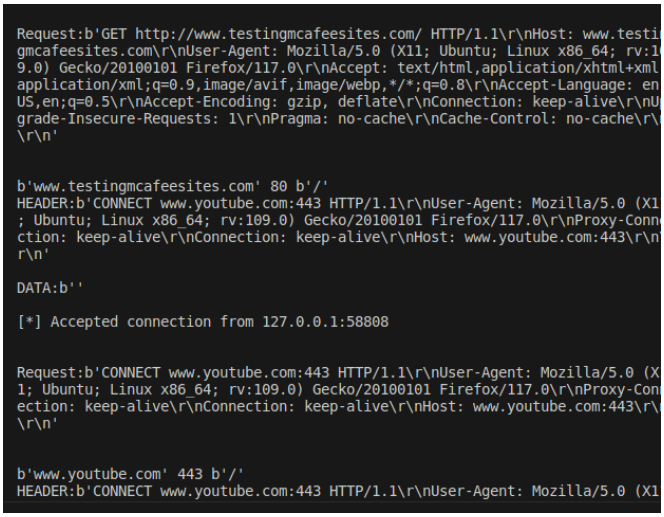
**(8) Browser communicates with remote host/server [example.com] via proxy (explicit)**

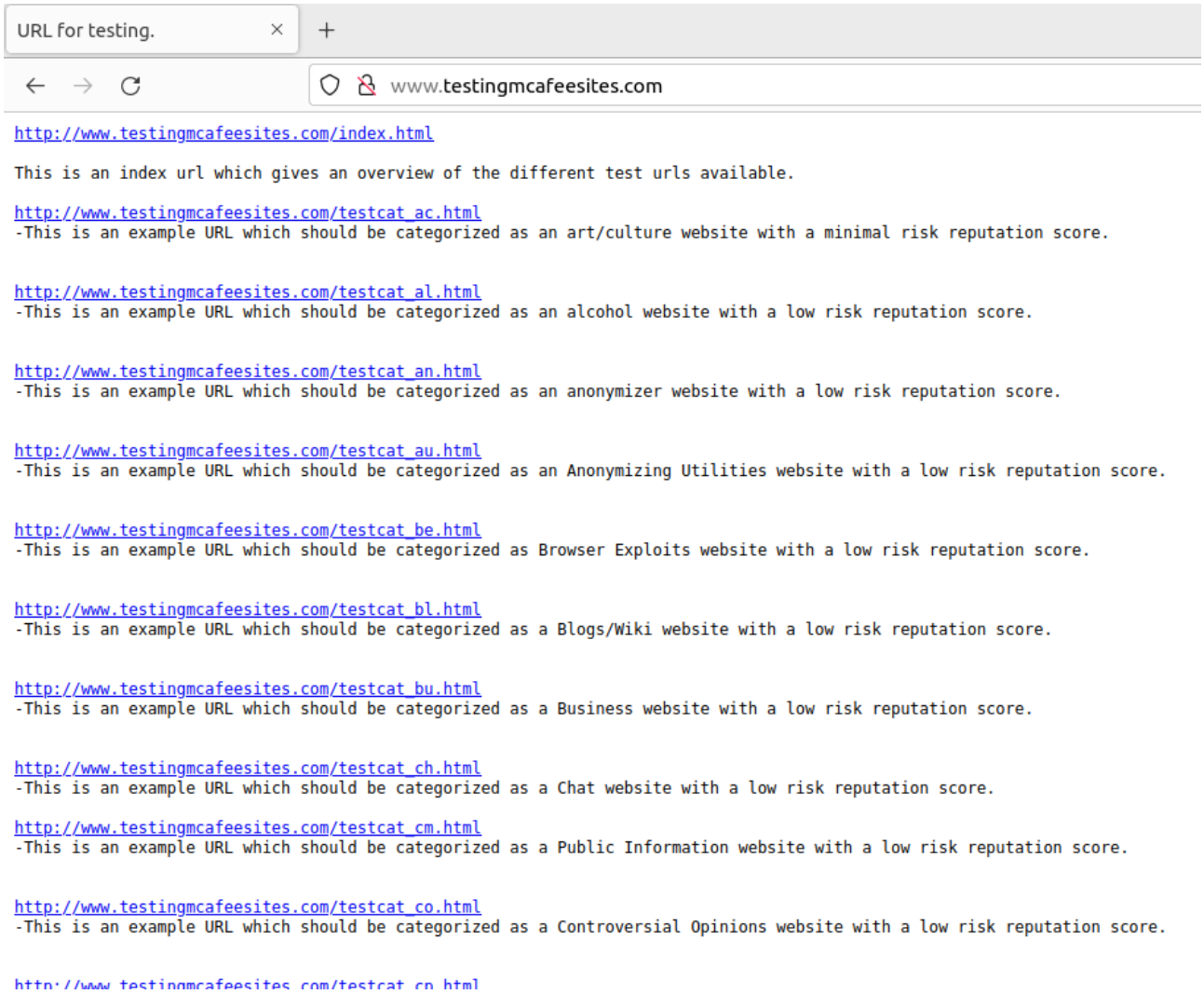



**(9) Browser communicates with remote host/server [example.com] via extended proxy (explicit)**

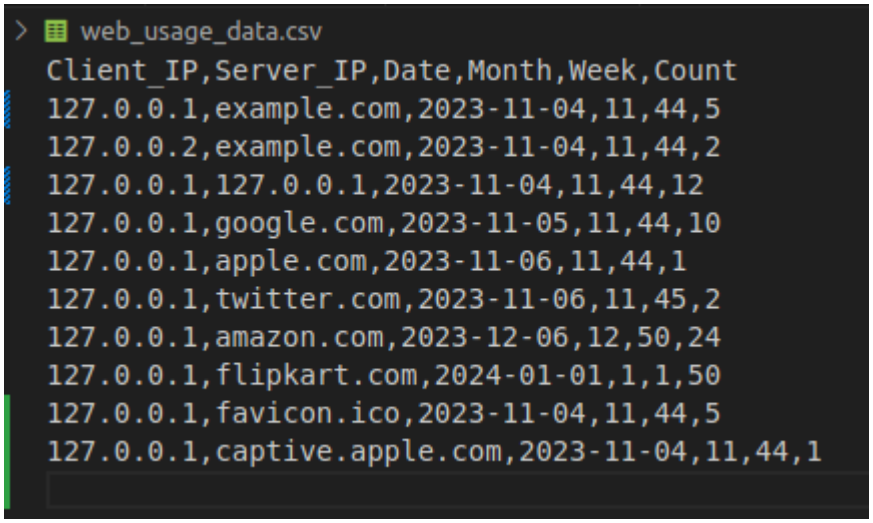

**(10) Browser communicates with remote host/server [example.com] via proxy (implicit)**



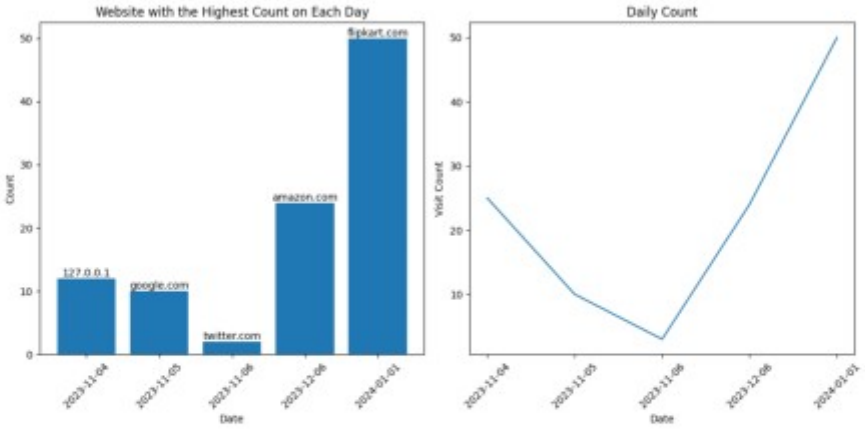





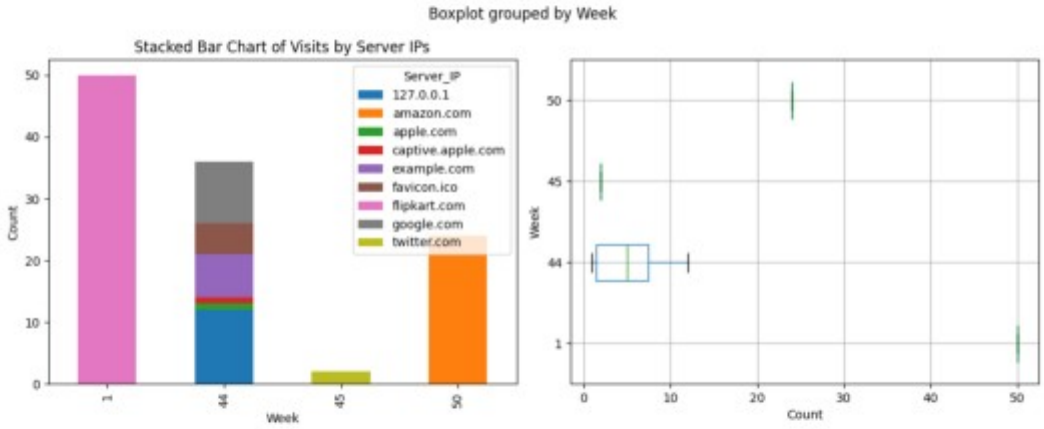

## (11) Web usage statistics



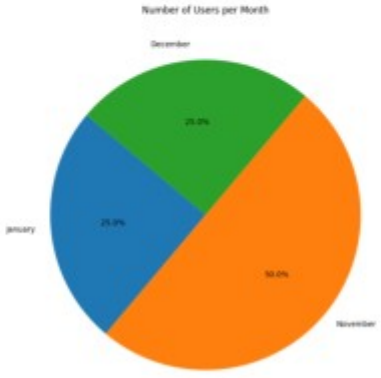
# Daily Statistics




# Weekly Statistics




# Monthly Statistics



# ANTI-PLAGIARISM Statement

I certify that this assignment/report is my own work, based on my personal study and/or research and that I 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. I 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 I have not copied in part or whole or otherwise plagiarized the work of other students and/or persons. Additionally, I acknowledge that I may have used AI tools, such as language models (e.g., ChatGPT, Bard), for assistance in generating and refining my assignment, and I 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, I understand my responsibility to report honour violations by other students if I become aware of it.

Name: Yug Patel, CS23MTECH14019
Somya jain, CS23MTECH12011
Anil Kumar, CS23MTECH13001

Date:05/11/2023
Signature: Yug Patel, CS23MTECH14019
Somya jain, CS23MTECH12011
Anil Kumar, CS23MTECH13001