

Programming Assignment 1 COEN 317 (Distributed Systems)

Submitted by: Arpita Verma

W1632653

Date: 7th Feb 2023

Objective: To build a functional Web Server

Table of Contents

Description of the Assignment	3
Details.....	3
List of Files	3
Instructions for running the Program	3
Screenshots of successful creation of the Web Server using port 8080 with incoming request logs	4
Screenshots of Successful Connections	5
1. Default page	5
2. HTML File Type.....	5
3. Accessing Index html from one folder called SCU	6
4. Accessing png image format.....	7
5. Accessing jpg image format.....	7
6. Accessing gif format	8
7. Accessing txt file type.....	9
Different Status Codes.....	10
1. Status code : 200 OK	10
2. Status code : 403 Forbidden	11
3. Status code: 404 NOT FOUND	12
4. Status code: 400 Bad Request	13
5. Status code: 500 Internal Server Error	14
References.....	15

Description of the Assignment

The goal of this programming assignment is to build a functional web server. The web server will be listening for the connections on a socket (bound to a specific port on the host machine).

Details

For this programming assignment, I am implementing a HTTP web server program which can support multi-threading also. It accepts and responds to HTTP requests.

The Web server will be listening for connection on a socket and the clients will be connecting to this socket and will try to retrieve files from the server.

The client sends request to the server and when the connection is established, a timeout of 3 secs is used for keeping the connection alive for 3 secs. The default file is index.html, so if the user do not enter and filename, the default page will be loaded.

This web server responds with appropriate responses based on the requests provided to it. The different types of responses shown here are 200, 400, 403, 404, 500.

The server file is coded using python programming.

The port used is 8080.

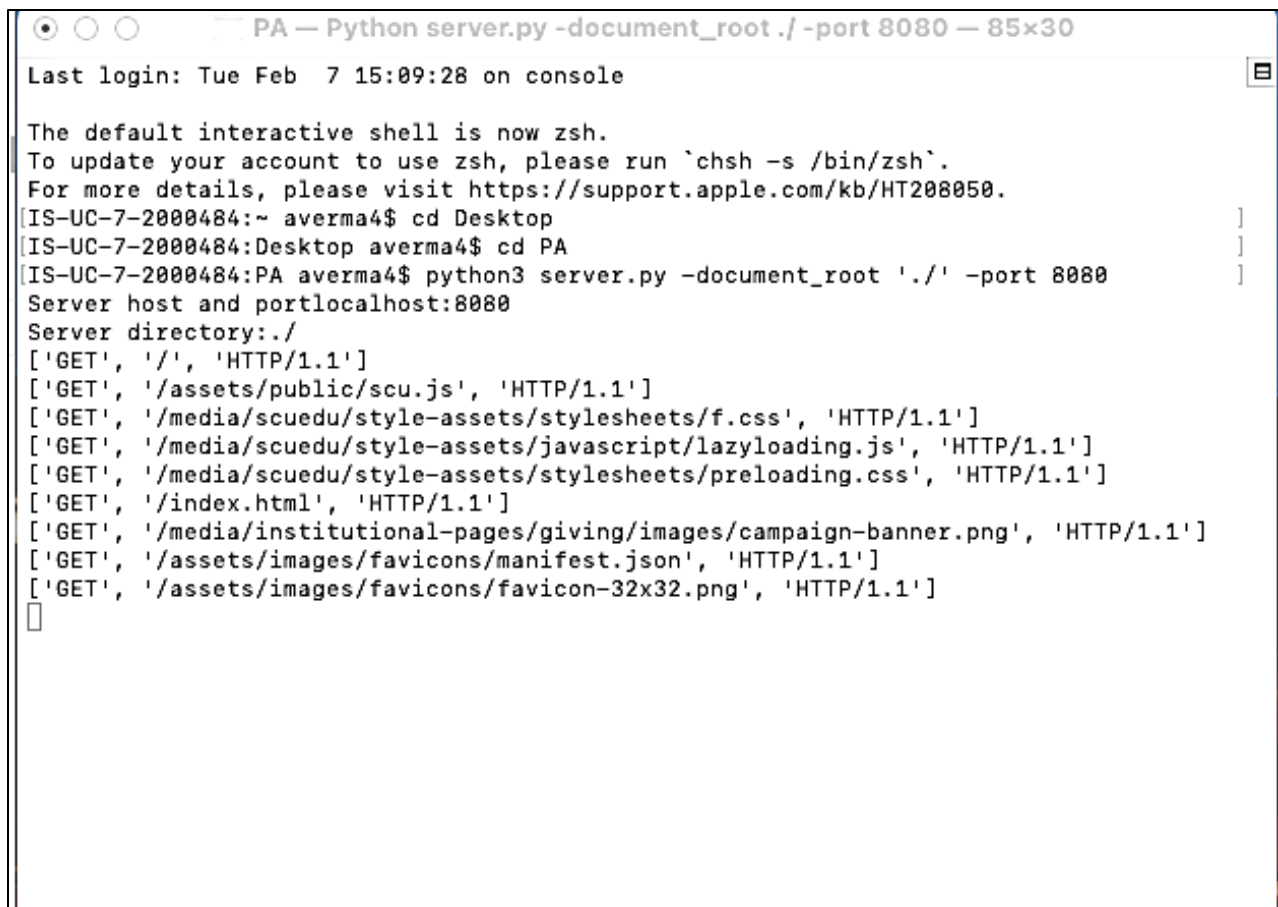
List of Files

1. Readme.pdf
2. Server.py
3. Index.html
4. Other files for testing purposes

Instructions for running the Program

1. Using the default browser – type **localhost:8080** in the address bar
Initially when the server code is not run, this command will not open anything in the browser.
2. To start the web server – run below command in the terminal of Mac/Linux OS –
python3 server.py -document_root './' -port 8080

Screenshots of successful creation of the Web Server using port 8080 with incoming request logs



```
PA — Python server.py -document_root ./ -port 8080 — 85x30
Last login: Tue Feb  7 15:09:28 on console

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
[IS-UC-7-2000484:~ averma4$ cd Desktop
[IS-UC-7-2000484:Desktop averma4$ cd PA
[IS-UC-7-2000484:PA averma4$ python3 server.py -document_root './' -port 8080
Server host and portlocalhost:8080
Server directory:./
['GET', '/', 'HTTP/1.1']
['GET', '/assets/public/scu.js', 'HTTP/1.1']
['GET', '/media/scuedu/style-assets/stylesheets/f.css', 'HTTP/1.1']
['GET', '/media/scuedu/style-assets/javascript/lazyloading.js', 'HTTP/1.1']
['GET', '/media/scuedu/style-assets/stylesheets/preloading.css', 'HTTP/1.1']
['GET', '/index.html', 'HTTP/1.1']
['GET', '/media/institutional-pages/giving/images/campaign-banner.png', 'HTTP/1.1']
['GET', '/assets/images/favicons/manifest.json', 'HTTP/1.1']
['GET', '/assets/images/favicons/favicon-32x32.png', 'HTTP/1.1']
█
```

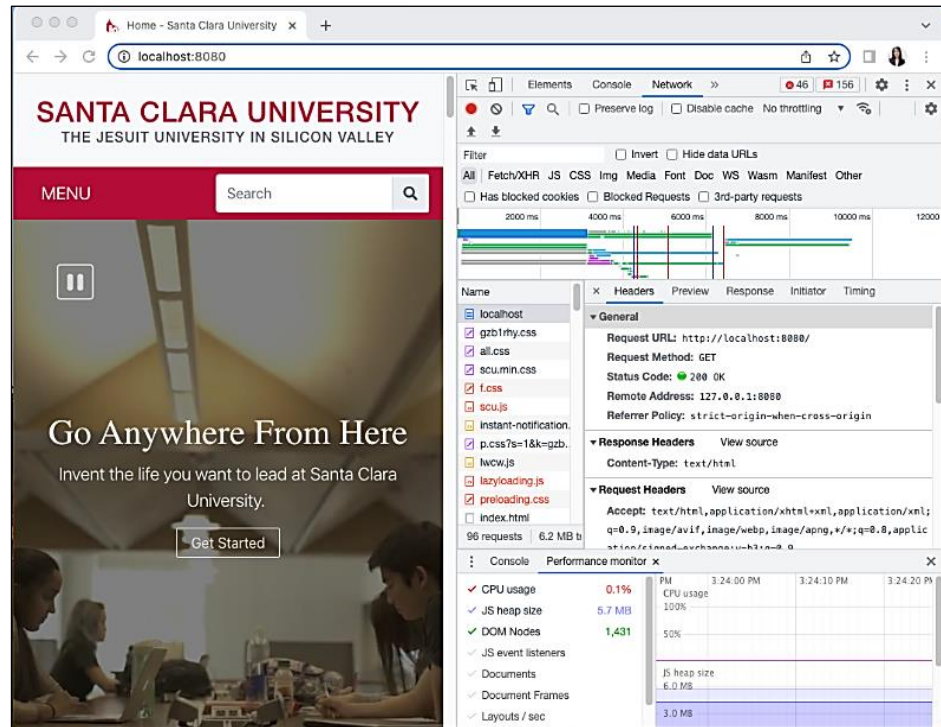
Figure 1Web Server Creation using port 8080 with incoming logs

Screenshots of Successful Connections

1. Default page

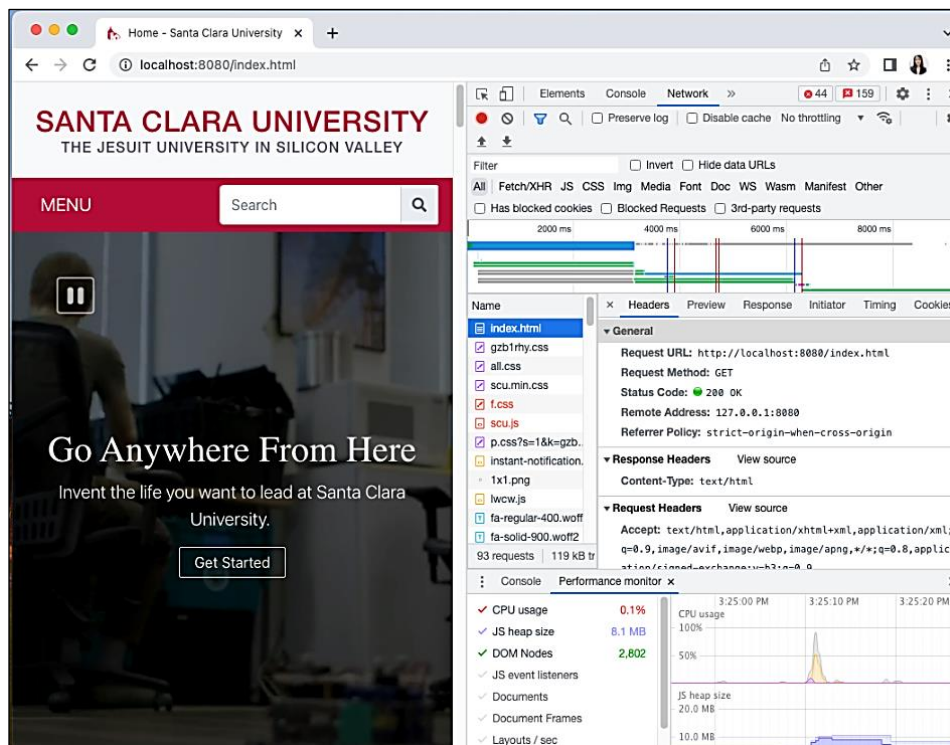
URL:

localhost:8080



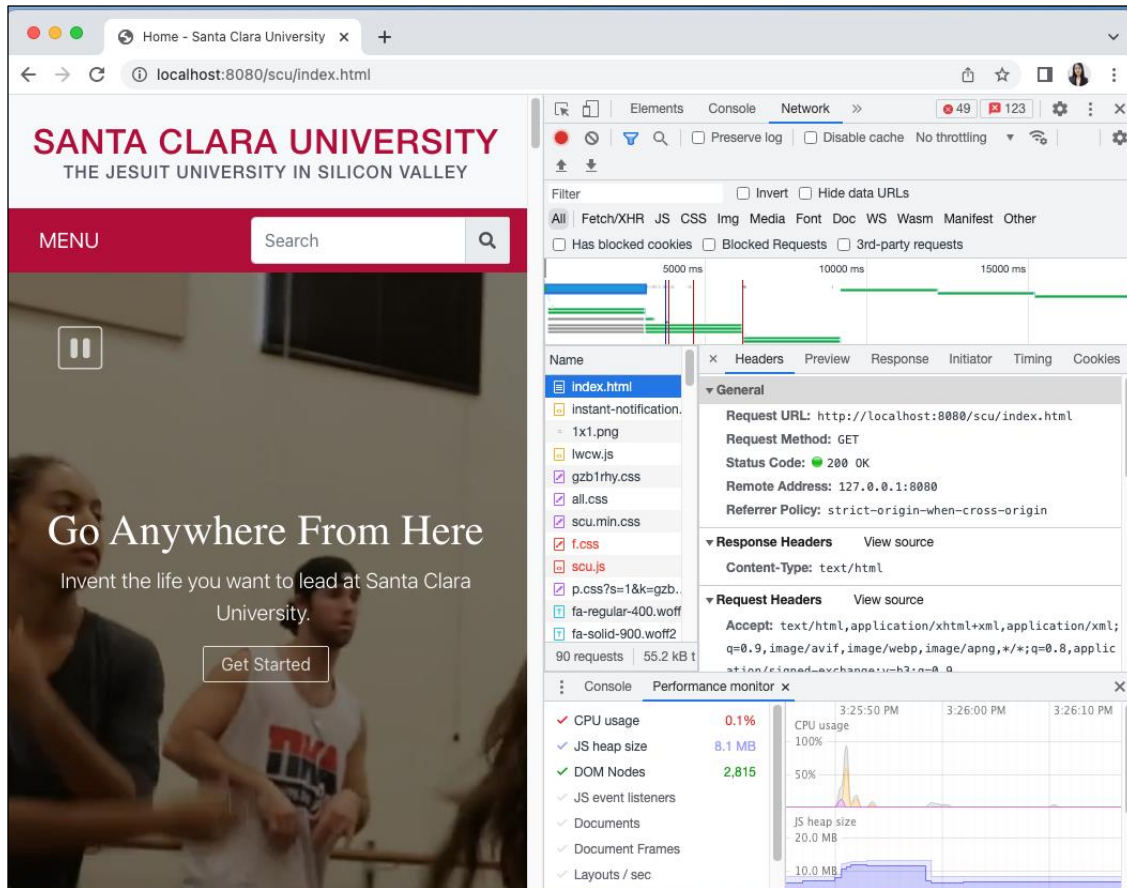
2. HTML File Type

URL :**localhost:8080/index.html**



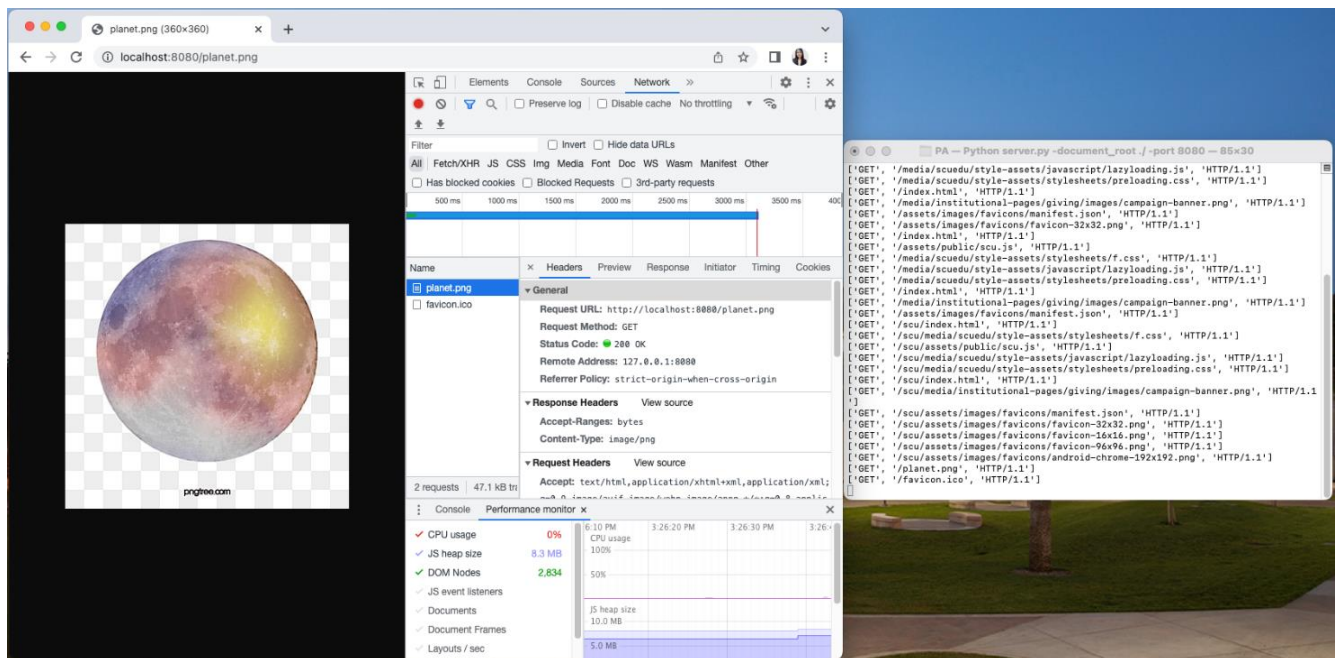
3. Accessing Index html from one folder called SCU

URL: **localhost:8080/SCU/index.html**



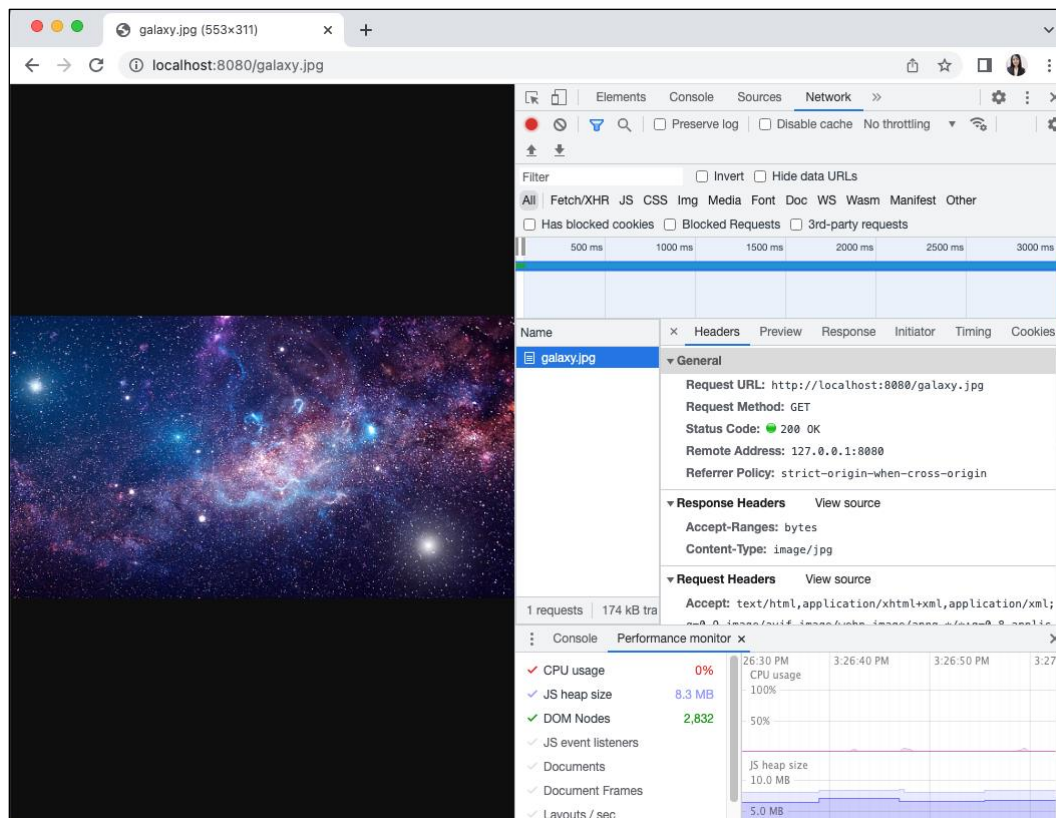
4. Accessing png image format

URL: **localhost:8080/planet.png**



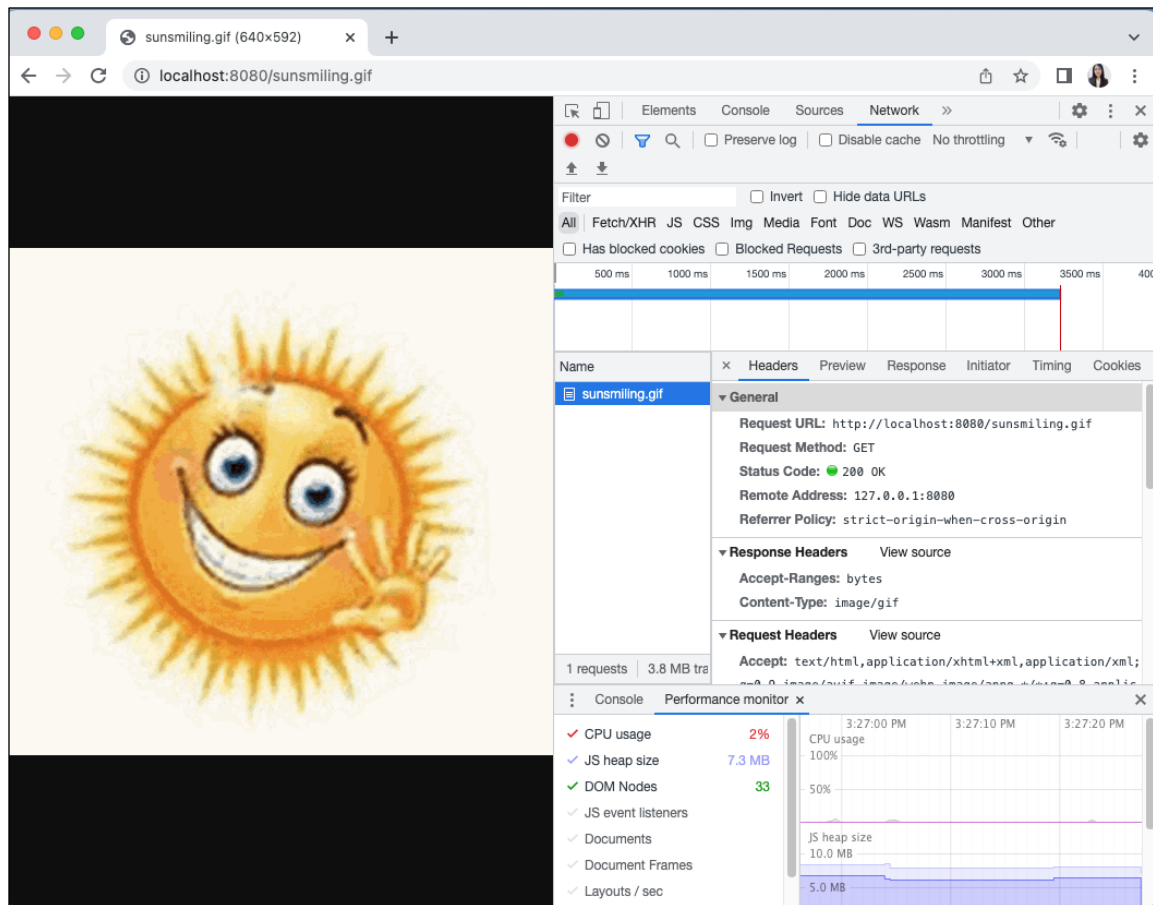
5. Accessing jpg image format

URL: **localhost:8080/galaxy.jpg**



6. Accessing gif format

URL: `localhost:8080/sunsmiling.gif`



7. Accessing txt file type

URL: **localhost:8080/PAtextfile.txt**

Programming Assignment 1 Distributed Systems (Winter 2023) To build a functional web server - Test Text File Sample Text File to check the working of the webs= server response to a text file

The screenshot shows a web browser window with the address bar displaying `localhost:8080/PAtextfile.txt`. The page content is a text file with the text: "Programming Assignment 1 Distributed Systems (Winter 2023) To build a functional web server - Test Text File Sample Text File to check the working of the webs= server response to a text file".

The browser's developer tools are open, showing the Network tab. A single request is listed: `PAtextfile.txt`. The request details are as follows:

- General:**
 - Request URL: `http://localhost:8080/PAtextfile.txt`
 - Request Method: `GET`
 - Status Code: `200 OK`
 - Remote Address: `127.0.0.1:8080`
 - Referrer Policy: `strict-origin-when-cross-origin`
- Response Headers:**
 - `Content-Type: text/html`
- Request Headers:**
 - `Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=h3;q=0.0`

The Performance monitor shows the following metrics:

- CPU usage: 0%
- JS heap size: 2.8 MB
- DOM Nodes: 16
- JS event listeners: 0
- Documents: 1
- Document Frames: 1
- Layouts / sec: 0

Different Status Codes

1. Status code : 200 OK

We get this status code when everything runs successfully.

URL : **localhost:8080**

The screenshot shows a web browser window displaying the Santa Clara University homepage. The browser's address bar shows the URL `localhost:8080`. The page content includes the university's name, a search bar, and a large banner with the text "Go Anywhere From Here" and "Invent the life you want to lead at Santa Clara University." A "Get Started" button is visible at the bottom of the banner.

The browser's developer tools are open, showing the Network and Performance monitors. The Network monitor displays a list of requests, including `localhost`, `gzb1rhy.css`, `all.css`, `scu.min.css`, `f.css`, `scu.js`, `gzb1rhy.css`, `p.css?s=1&k=gzb`, `instant-notificatio`, `lwcw.js`, `lazyloading.js`, and `evolving.ne`. The selected request is `localhost`, which has a status code of 200 OK. The Performance monitor shows a timeline of the page load, with a peak in CPU usage and JS heap size around 3:28:40 PM.

2. Status code : 403 Forbidden

We get Forbidden status code of 403 when we try to access video file with extension of .mp4 because .mp4 files are restricted in this scenario.

URL : **localhost:8080/video.mp4**

The screenshot shows a web browser at the URL `localhost:8080/video.mp4`. The main content area displays a message: "Access to localhost was denied. You don't have authorization to view this page. HTTP ERROR 403". A "Reload" button is visible at the bottom right of the page.

The Chrome DevTools interface is open, showing the Network and Performance panels.

Network Panel:

- Filter: ☐ Invert ☐ Hide data URLs
- All ☐ Fetch/XHR ☐ JS ☐ CSS ☐ Img ☐ Media ☐ Font ☐ Doc ☐ WS ☐ Wasm ☐ Manifest ☐ Other
- ☐ Has blocked cookies ☐ Blocked Requests ☐ 3rd-party requests
- Timeline: 10000 ms, 20000 ms, 30000 ms, 40000 ms
- Name:
 - ☐ collect?r=uc&app=...
 - ☒ tr/
 - ☐ collect?v=1&_v=j...
 - ☐ 521
 - ☐ 522
 - ☒ FeaturedStory-Ba...
 - ☐ manifest.json
 - ☐ collect?v=2&tid=...
 - ☒ video.mp4
 - ☐ data:image/png;b...
 - ☐ data:image/png;b...
 - ☐ data:image/png;b...
- 98 requests | 81.5 kB

Headers Panel (selected for video.mp4):

- General
 - Request URL: `http://localhost:8080/video.mp4`
 - Request Method: GET
 - Status Code: **403 Forbidden**
 - Remote Address: `127.0.0.1:8080`
 - Referrer Policy: `strict-origin-when-cross-origin`
- Request Headers
 - 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`
 - Accept-Encoding: `gzip, deflate, br`
 - Accept-Language: `en-US,en;q=0.9`
- View source

Performance Panel:

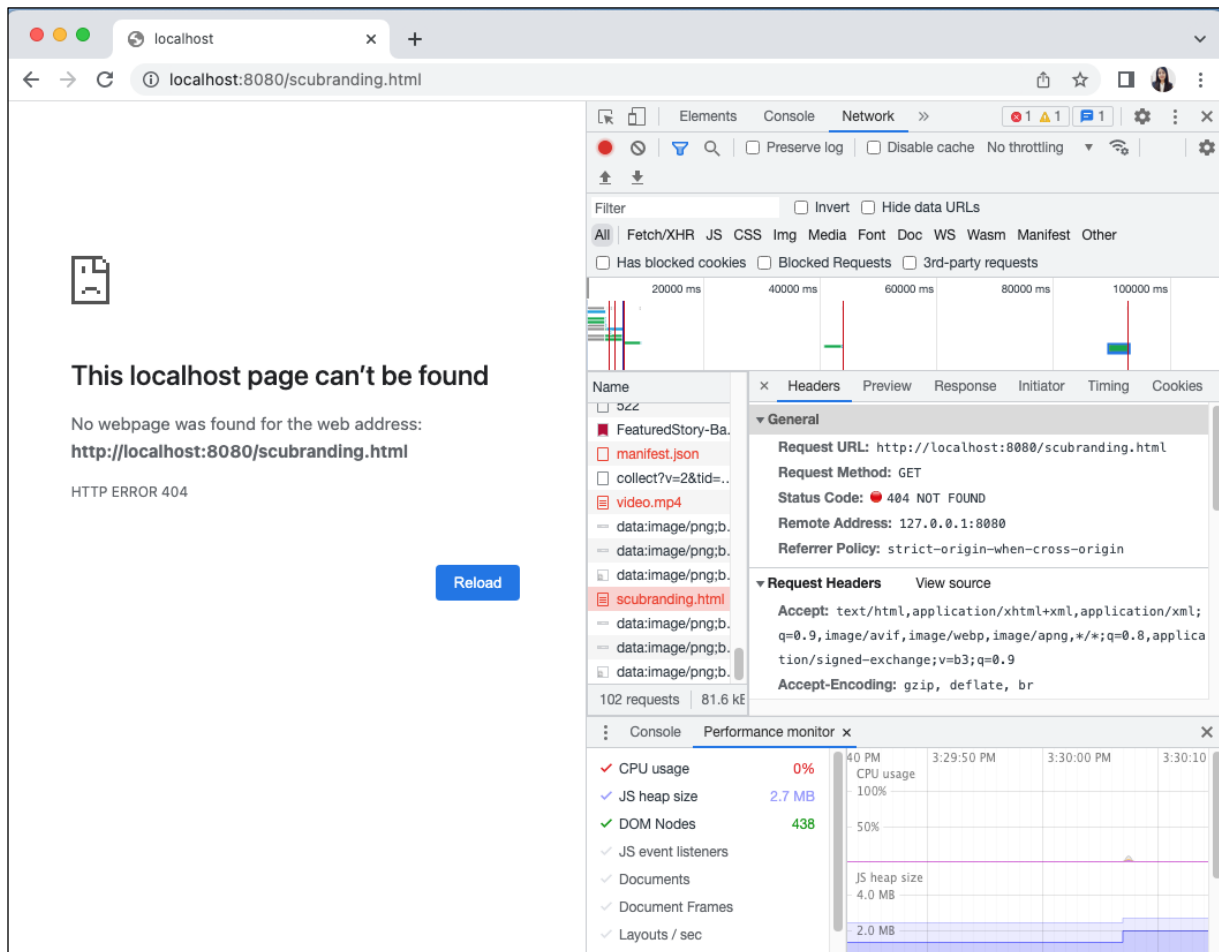
- CPU usage: 0%
- JS heap size: 2.4 MB
- DOM Nodes: 219
- JS event listeners
- Documents
- Document Frames
- Layouts / sec

The Performance panel also shows a timeline graph with CPU usage (0% to 100%) and JS heap size (3.0 MB to 6.0 MB) over time (3:29:10 PM to 3:29:30 PM).

3. Status code: **404 NOT FOUND**

We get this status code when the file or URL that is being accessed is not present on the server.

URL: **localhost:8080/scubranding.html**



4. Status code: 400 Bad Request

We get status code when we there is any special character like * or ! in the URL.

URL: **localhost:8080/***

The screenshot shows a web browser at `localhost:8080/*` displaying an error page. The page content includes a sad face icon, the text "This page isn't working", a message "If the problem continues, contact the site owner.", and "HTTP ERROR 400". A "Reload" button is visible.

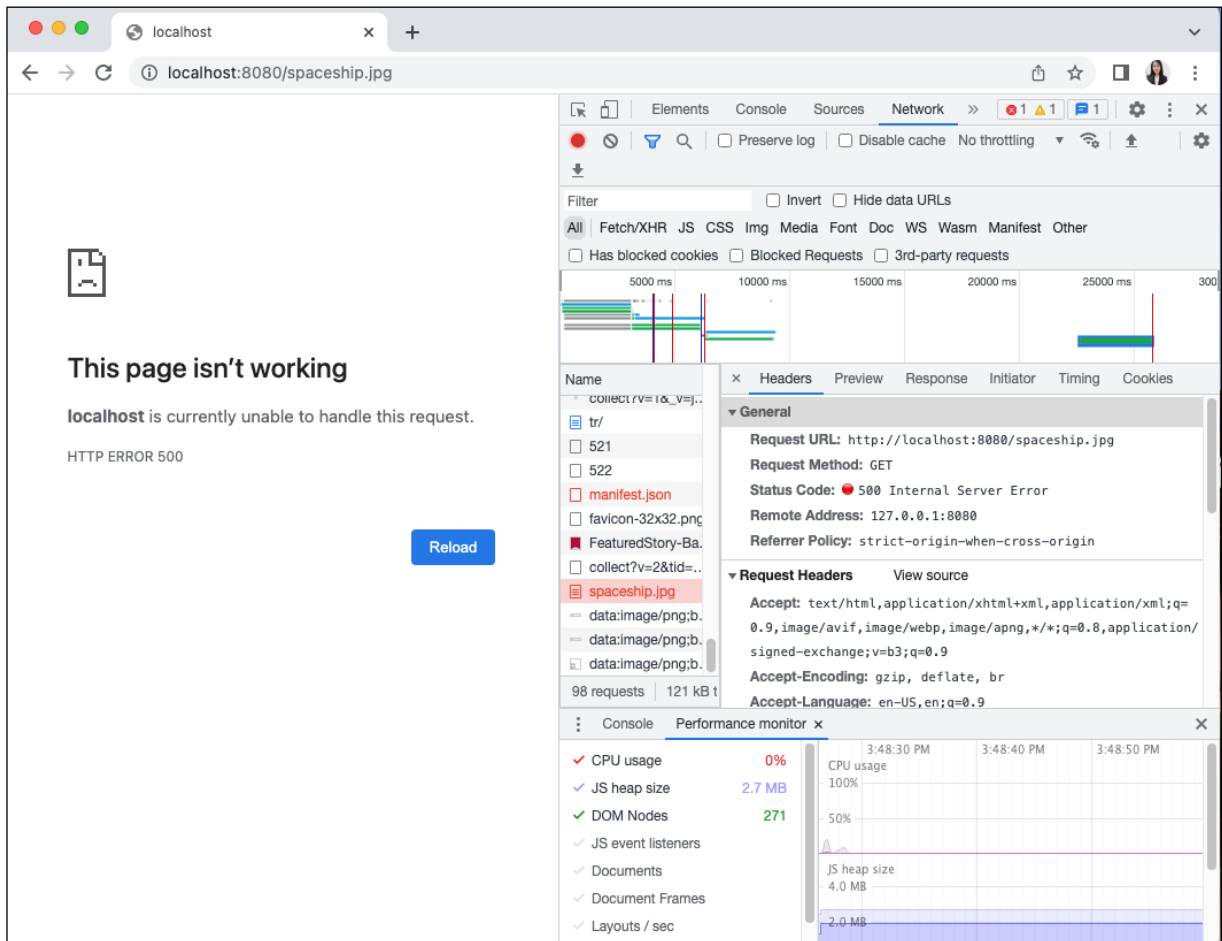
The Chrome DevTools Network panel is open, showing a list of requests. The selected request is a GET request to `http://localhost:8080/*` with a status code of 400 Bad Request. The request headers are visible, showing `Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9` and `Accept-Encoding: gzip, deflate, br`.

The Performance panel is also open, showing a timeline of CPU usage, JS heap size, DOM Nodes, JS event listeners, Documents, Document Frames, and Layouts / sec. The CPU usage is 0%, JS heap size is 2.4 MB, and DOM Nodes are 207.

5. Status code: 500 Internal Server Error

We get this status code when file has no read permissions. Here spaceship.jpg has no read permissions given.

URL: **local:8080/spaceship.jpg**



References

1. Wget installation Tutorial - <https://www.youtube.com/watch?v=Oe40pxcjPpM>
2. Socket Programming basics - <https://www.geeksforgeeks.org/socket-programming-python/>
3. Python library for HTTP servers - <https://docs.python.org/3/library/http.server.html>
4. SCU Website - <https://www.scu.edu/>