

Report

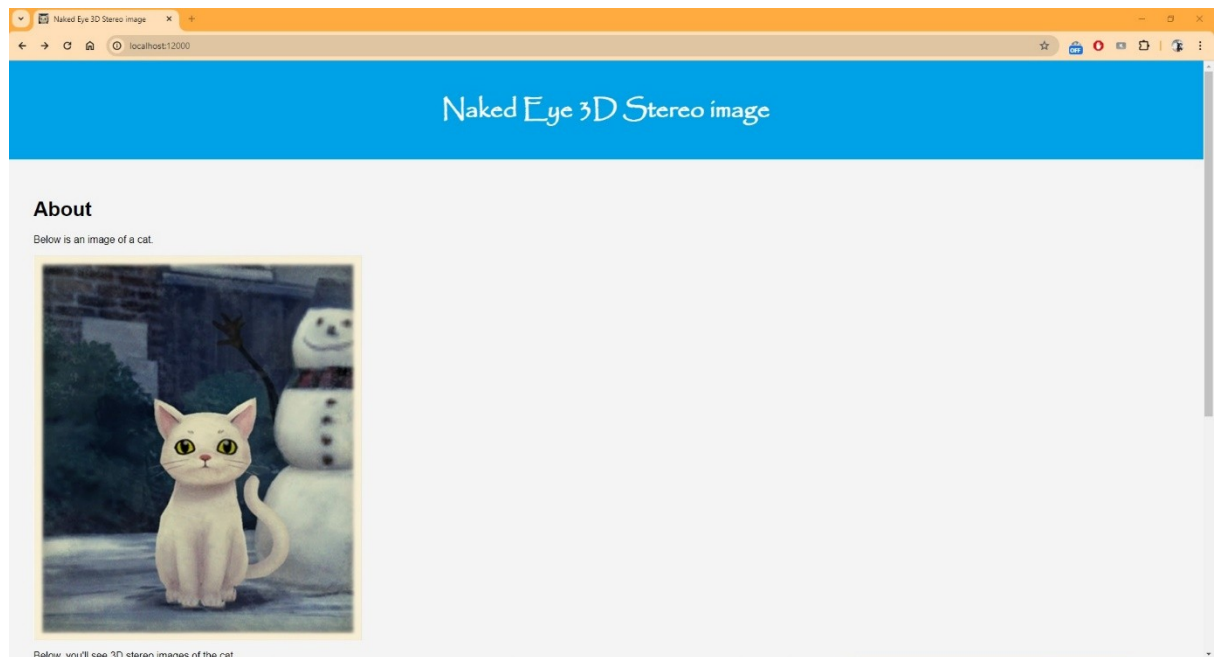


Figure 1: Browser Screenshot

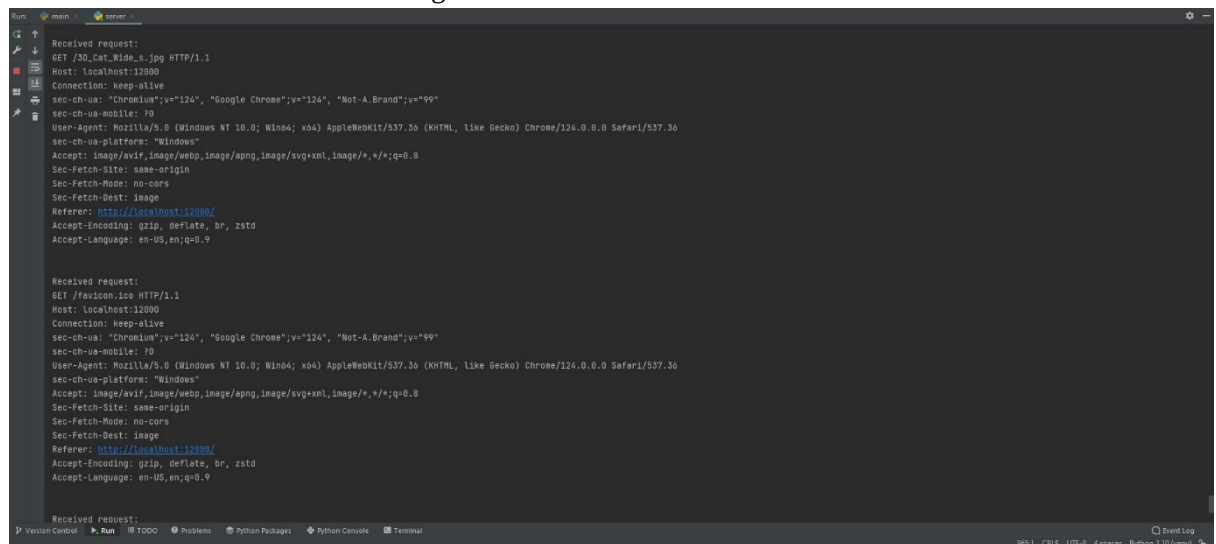
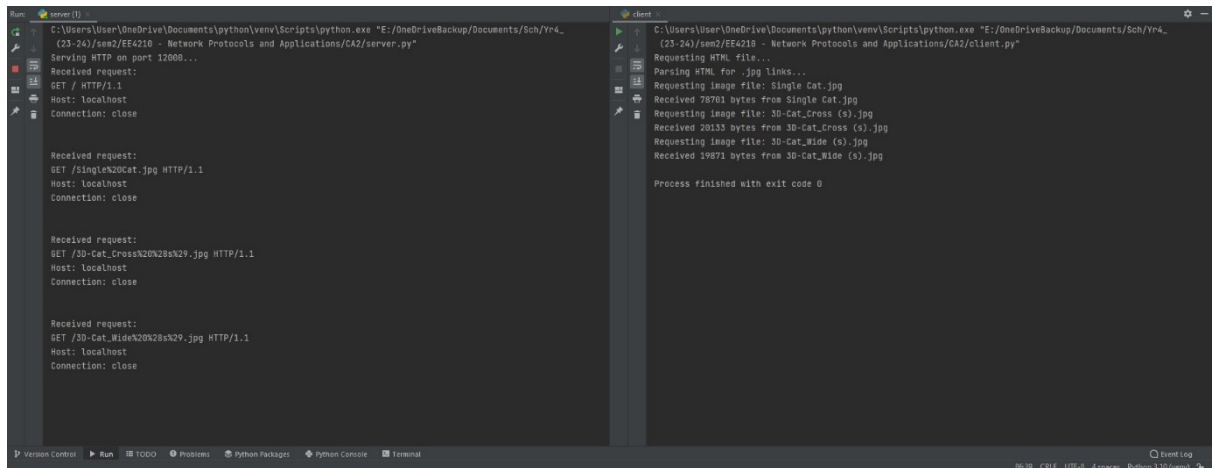


Figure 2: Server Terminal upon request



```
server (1)
C:\Users\User\OneDrive\Documents\python\venv\Scripts\python.exe "E:/OneDriveBackup/Documents/Sch/Yr4_
(23-24)/sem2/EE4210 - Network Protocols and Applications/CA2/server.py"
Serving HTTP on port 12000...
Received request:
GET / HTTP/1.1
Host: localhost
Connection: close

Received request:
GET /SingleCat.jpg HTTP/1.1
Host: localhost
Connection: close

Received request:
GET /3D-Cat_Cross%20%28s%29.jpg HTTP/1.1
Host: localhost
Connection: close

Received request:
GET /3D-Cat_Wide%20%28s%29.jpg HTTP/1.1
Host: localhost
Connection: close

client
C:\Users\User\OneDrive\Documents\python\venv\Scripts\python.exe "E:/OneDriveBackup/Documents/Sch/Yr4_
(23-24)/sem2/EE4210 - Network Protocols and Applications/CA2/client.py"
Requesting HTML file...
Parsing HTML for .jpg links...
Requesting image file: Single Cat.jpg
Received 78701 bytes from Single Cat.jpg
Requesting image file: 3D-Cat_Cross (s).jpg
Received 20133 bytes from 3D-Cat_Cross (s).jpg
Requesting image file: 3D-Cat_Wide (s).jpg
Received 19871 bytes from 3D-Cat_Wide (s).jpg

Process finished with exit code 0
```

Figure 3: Client and Server Terminal when client sends request

Improvements

To enhance the speed at which images are downloaded in the client, multithreading can be implemented on both the server and client sides. On the server, multithreading allows multiple requests to be handled simultaneously, improving the server's ability to serve multiple files concurrently to different clients or the same client making multiple requests. For the client, using multithreading to request and download multiple images at the same time can reduce the total download time by leveraging parallel processing, rather than waiting for each download to complete sequentially.