**IMPLEMENTATION OF HTTP WEB PROXY SERVER**

*Report submitted to the SASTRA Deemed to be University*

*as the requirement for the course*

**CSE302: COMPUTER NETWORKS**

*Submitted by*

**ABHIMANYU SINGH**

**(Reg. No.: 124015001, B.Tech – Information Technology)**

**December 2022**



**SCHOOL OF COMPUTING**

**THANJAVUR – 613 401**



**SCHOOL OF COMPUTING**

**THANJAVUR – 613 401**

**Bonafide Certificate**

This is to certify that the report titled “**Implementation of HTTP Web Proxy Server”**

submitted as a requirement for the course, **CSE302: COMPUTER NETWORKS** for B.Tech.

is a bonafide record of the work done by **Shri. Abhimanyu Singh (Reg. No.: 124015001,**

**B.Tech – Information Technology)** during the academic year 2022-2023, in the School of

Computing.

Project Based Work *Viva* voce held on

**Examiner 1 Examiner 2**

**List of Figures**

|  |  |  |
| --- | --- | --- |
| **Figure No.** | **Title** | **Page No.** |
| Fig 1.1 | Proxy Server | 7 |
| Fig 1.2 | Forward And Reverse Proxies | 8 |
| Fig 1.3 | Clients requesting objects through a Web cache | 9 |
| Fig 1.4 | HTTP vs HTTP | 10 |
| Fig 1.5 | Working of a web proxy server | 11 |
| Fig 3.1 | Directory consisting of required files & “cached” folder for storing and retrieving Cache files. | 35 |
| Fig 3.2 | Firefox used to forward all its traffics to port number 6969 | 35 |
| Fig 3.3 | Proxy Server Menu | 36 |
| Fig 3.4 | Created files in directory | 36 |
| Fig 3.5 | HTTP GET Request | 37 |
| Fig 3.6 | Inspected website without caching (accessing first time) | 38 |
| Fig 3.7 | Cached file in local folder | 38 |
| Fig 3.8 | Cached copy found in directory and given to client. | 39 |
| Fig 3.9 | Inspected website after caching. | 39 |
| Fig 3.10 | Request blocked and list of blocked sites. | 40 |
| Fig 3.11 | Webpage unable to load due to forbidden access. | 41 |
| Fig 3.12 | Closing proxy server | 41 |

**Abbreviations**

FTP File Transfer Protocol

GUI Graphical User Interface

IP Internet Protocol

HTTP HyperText Transfer Protocol

HTTPS HyperText Transfer Protocol Secure

NAT Network Address Translation

SSL Secure Socket Layer

TCP Transfer Control Protocol

URL Uniform Resource Locator

**ABSTRACT**

The internet is used by billions of individuals in this developing world. Major problems occur everyday web pages, like security concerns, Response time and restricting sensitive content to the users.

Because so many people are using the internet at once, there is a rise in the intensity of traffic on the link that carries data between servers and clients. As a result, the average response time, or the amount of time it takes from the browser to request an object to receiving it, increases and will likely be in the minutes range, which is undesirable to some users. A Proxy Server can be used to implement Caching, reducing the amount of traffic and the delay, to solve this problem. resulting in a quicker response time. Through the use of CDN and the localization of much of the traffic, web caches are becoming an increasingly significant part of the internet.

Using a proxy server, one can also implement parental control (blocking sensitive websites).

**KEYWORDS:** Blocking websites, Caching, Traffic Intensity, Response Time, Delay.

**Table of Contents**

**Title Page No.**

Bonafide Certificate 1

List of Figures 3

Abbreviations 4

Abstract 5

1. Introduction, Merits and Demerits of Work 7-13

2. Source Code 14-34

3. Snapshots 35-41

4. Conclusion and Future Plans 42

5. References 43

**INTRODUCTION**

* 1. **PROXY SERVER:**

Proxy servers are typically employed as a gateway between a user and the internet. Implemented between online visitors' browsers and the websites they visit. A computer uses an IP address to connect to the internet. A proxy server is comparable to an internet-connected machine with a unique IP address. As a result, it aids in preventing online intruders from accessing a private network.

Network Address Translation, often known as NAT, is frequently combined with proxies to mask the client's IP address from the distant server.

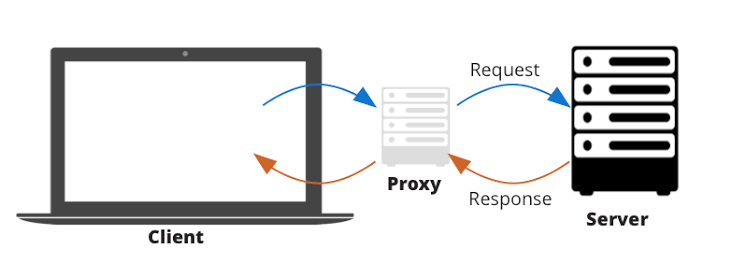


Fig 1.1. Proxy Server

**Types:**

**1.1.1 Application Level and Circuit Level:**

Application Level Proxy: A proxy that only allows access to a certain type of material, such as an HTTP Web Proxy or an FTP Proxy. They are committed to completing a particular task. Various applications can be supported by Circuit Level Proxies.

**1.1.2 Forward and Reverse Proxies:**

Forward Proxies are proxies located near the client which conceals details of the client from the server and the remote server will not know the true identity of the requesting client whereas Reverse proxies, on the other hand, are proxies that are located close to the server but conceal information about the server from clients. This prevents the remote server from knowing the genuine identity of the client making the request.



Fig 1.2. Forward And Reverse Proxies

**1.1.3 Other Proxies:**

These are the anonymous proxy servers that are accessible online and help to conceal users' identity.

* 1. **PROXY FUNCTIONS:**
     1. **Web Caching:**

A Web Caching Proxy Server keeps copies of previously accessed requested objects in this storage, which is part of the server's own disc storage. All HTTP queries can be set up in the user's web browser to go through the Web Proxy first. The procedures that follow in order to complete the transmission take place once these requests are directed to the web proxy server:

1. The browser connects over TCP to the Web cache and issues an HTTP request to the Web cache for the object.

2. Web caching checks to see if a duplicate of the object is locally saved. If a copy is present, the Web cache sends the item to the client browser as part of an HTTP response message.

3. The proxy initiates a TCP connection to the origin server if the web caching proxy does not already hold the object. The Web cache then uses the proxy-to-server TCP connection to send an HTTP request for the object. The origin server delivers the object along with an HTTP response to the Web cache after receiving this request.

4. After receiving the item, the web caching proxy makes a copy and stores it locally before sending a copy and an HTTP response message to the client browser via the already established TCP connection between cache and client.

Multiple clients using the proxy maximises the proxy cache's benefits, allowing other clients to access pages that have been cached by one client while experiencing dramatically reduced response times.

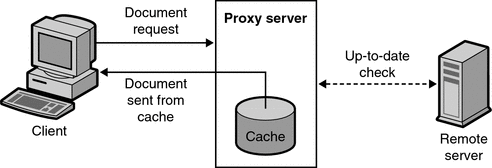


Fig 1.3. Clients requesting objects through a Web cache

* + 1. **Access Controls**

Web proxies are highly helpful in Private Networks, such as those found in Institutions, to prohibit users from accessing specific websites or even for parental control because they can give Access Control. A web proxy can block websites on a client's request if it is given a list of websites to ban. When a client request is examined, the proxy checks the received destination URL address against the list of prohibited websites and refuses access if they match.

* + 1. **Address Translation:**

When used in conjunction with NAT, web proxies hide the client's IP address by changing it to an external IP and making requests on their behalf.

**1.3. HTTPS Requests:**

SSL or TLS are used for HTTPS connections. In this case, the client and server exchange encrypted data. To ensure secure transactions, this is widely employed in the financial sector and is currently present practically everywhere.

However, this creates a problem for proxy servers because they have access to all the information or services that clients have requested to the server and there is no secure channel of communication between the two. Since these proxy servers cache all the recently accessed web pages and their traffic, there is a higher risk of a Man In The Middle Attack.

However, these proxy servers will be unable to block or cache traffic of a certain kind known as HTTPS traffic.

Here, the proxy creates a secure connection between the client and the server instead of caching the objects. Implementation of this uses HTTP CONNECT Tunneling.

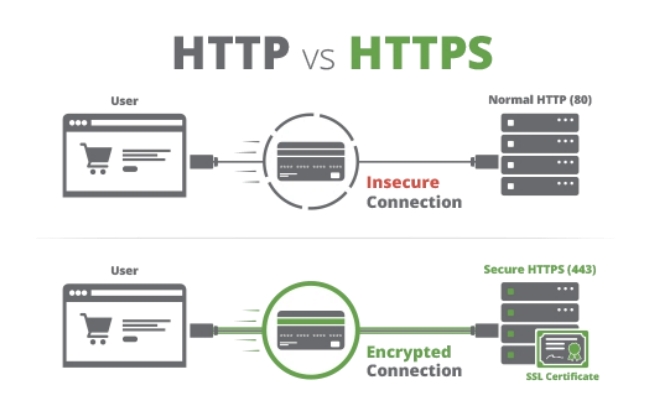


Fig 1.4. HTTP vs HTTPS

**Setting up and Running Proxy Server:**

A web page can be requested from the browser after the Proxy server programme has been launched from a cmd prompt. Utilizing the IP address and port number, the requests are routed to the proxy server (to which the proxy server programme is listening to).

The localhost can be changed to the IP address of the machine on which the proxy server code is executing if you want to utilise the proxy server with a browser and proxy on separate computers. Additionally, the port number must match the one that the proxy server is listening on.

IP Address used here: localhost

Port number: 6969

**1.4. WORKING**

Traffic from the client is transmitted to the proxy server, which then requests the requested data from the distant server on the client's behalf. The required data is subsequently forwarded to the client via the proxy server. This is helpful if the administrator wants to specify how the user should interpret the data. Clients won't be able to access particular websites, for instance, if the proxy server blocks them dynamically. The client can specify the website url to block, and access to that website will subsequently be limited. In the case of parental control, this aids in limiting the content that users or kids can see.

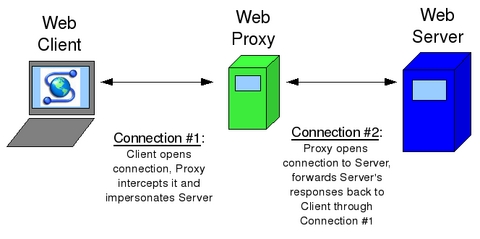


Fig 1.5. Working of a web proxy server

**1.4.1. The Implementation**

Implemented using java its TCP socket libraries. Here, Firefox was used to set up all of its traffic to the specified port (6969) and IP address(localhost), which were then used in the proxy configuration.

Main components of implementation :-

1. Proxy class

2. RequestHandler class.

**Proxy Class:**

In order to receive incoming socket connections from the client, the Proxy Class produces a Server Socket. As the server must serve numerous clients at once, the implementation ought to be multithreaded. As a result, the Proxy accepts the socket connection as it happens and starts a new thread to handle the request (RequestHandler). Multiple clients can have their requests served simultaneously thanks to the server's ability to accept new socket connections before the request has been entirely processed.

Additionally, blocking and caching functions are implemented in this case using a proxy server. It caches websites requested by clients and blocks specific websites listed in the block sites file in the directory.

It is suggested to save references to currently banned and cached websites in a data structure with a constant order search time because response time is crucial for the proxy server (Hashmap). If the file is not in the cache, there is very little overhead, and if the file is located in the cache and saved in a directory, there is an improvement in performance.

**The RequestHandler Class:**

It services the requests that come through to the proxy.

1.HTTP GET requests.

2. HTTP GET requests for files contained in the cache.

3. HTTPS CONNECT Requests.

**HTTP GET**

Standard request made when client tries to load a webpage.

1.Parse out the URL associated with the request.

2.Create a HTTP connection to this URL.

3.Echo the client’s GET request to the remote server.

4.Echo the server’s response back to the client and save a copy of file into the directory as proxy’s cache.

**HTTP GET for File in Cache**

Typical requests made by clients, in this case, the file is contained in the proxy’s cache.

1.Parse out the URL associated with the request

2.Hash the URL and use this as the key for the HashMap data structure.

3.Open file found to read.

4.Echo the contents of the file back to the client.

5.Close file.

**HTTPS - CONNECT Requests**

Since the CONNECT request uses ordinary HTTP, it is not encrypted. It includes the client's HTTPS request's destination address, which can be accessed through a proxy. Tunneling over HTTP.

1. Client sends CONNECT Request

2. The proxy extracts the destination URL.

3.Proxy establishes a connection using a normal socket to the remote server indicated by the URL.

4.If the proxy is successfully constructed, it notifies the client with a "Connection Established" (200) response, allowing the client to transfer the encrypted data to the proxy.

5. The proxy then transmits any data that is sent or received from a client to a distant server or vice versa.

Since all of this data will be encrypted, the proxy is unable to cache or even decrypt it.

**1.5. MERITS**

1. The proxy server can cache HTTP requests to speed up access to previously accessed web pages and reduce waiting times.

2. Access to HTTP websites may be blocked or limited.

3. CONNECT Tunneling, which offers secure connections and data encryption, allows HTTPS connections even if the server is an HTTP proxy server (SSL). Data security is increased, guarding against Man in the Middle attacks.

4. Offers safer Internet browsing and improved network performance.

**1.6. DEMERITS :**

1. Because the connection is made over CONNECT and the data is encrypted, HTTPS requests cannot be cached.

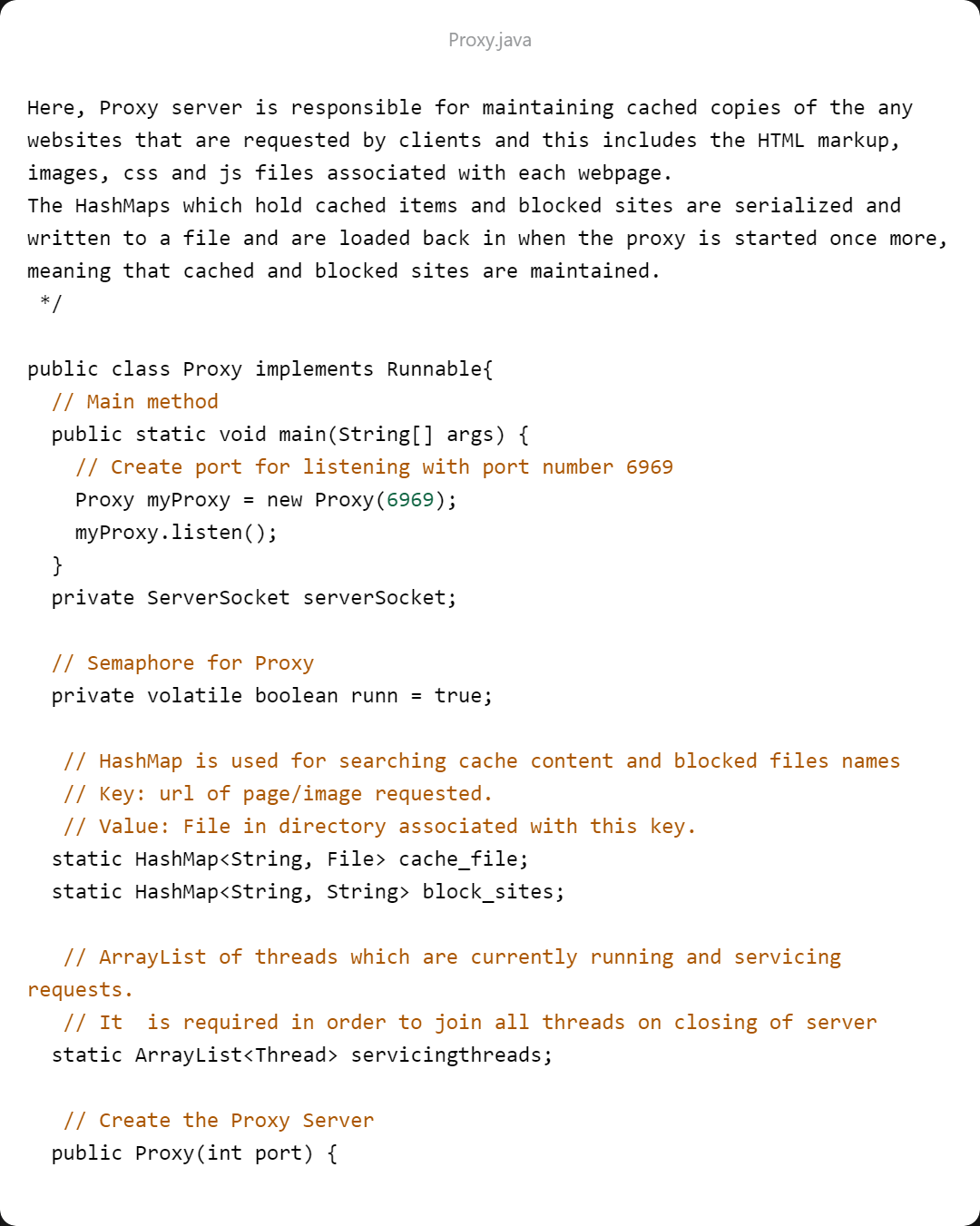
2. Users cannot block or limit access to HTTPS sites.

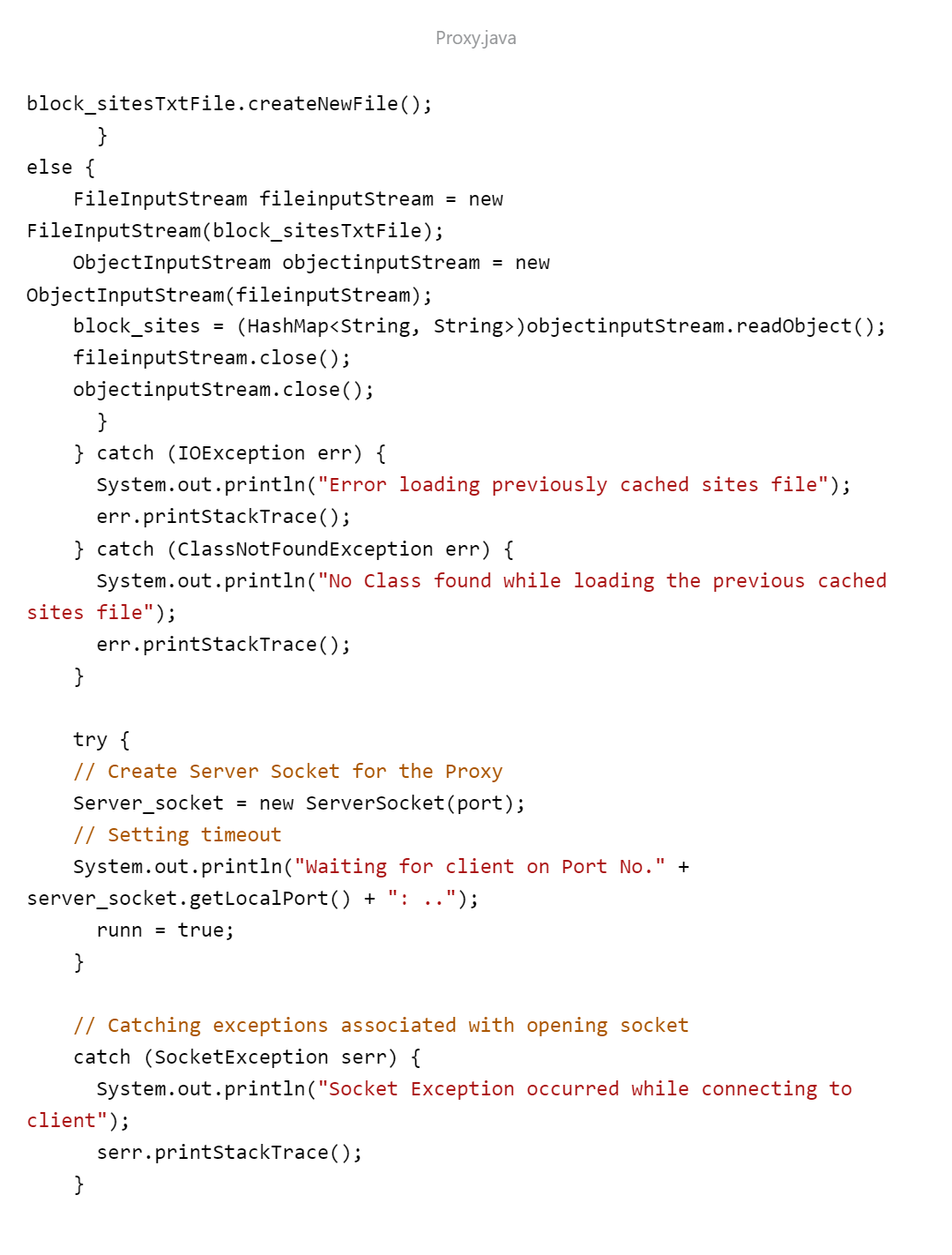
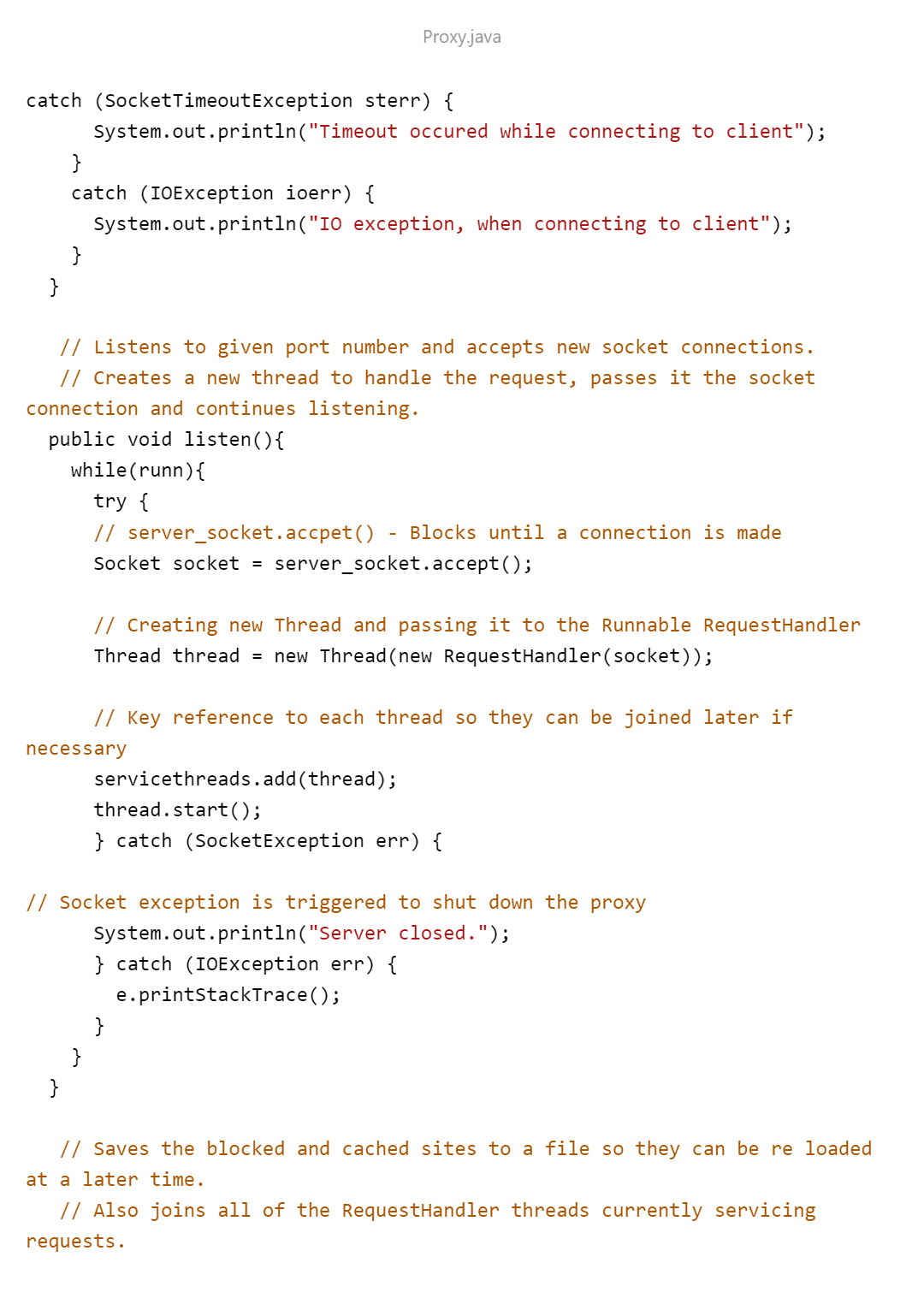
3. HTTP queries are not secure because they are not encrypted.

4. Adequate security precautions must be performed before transmission because using a proxy server is often like allowing a third party to access data.

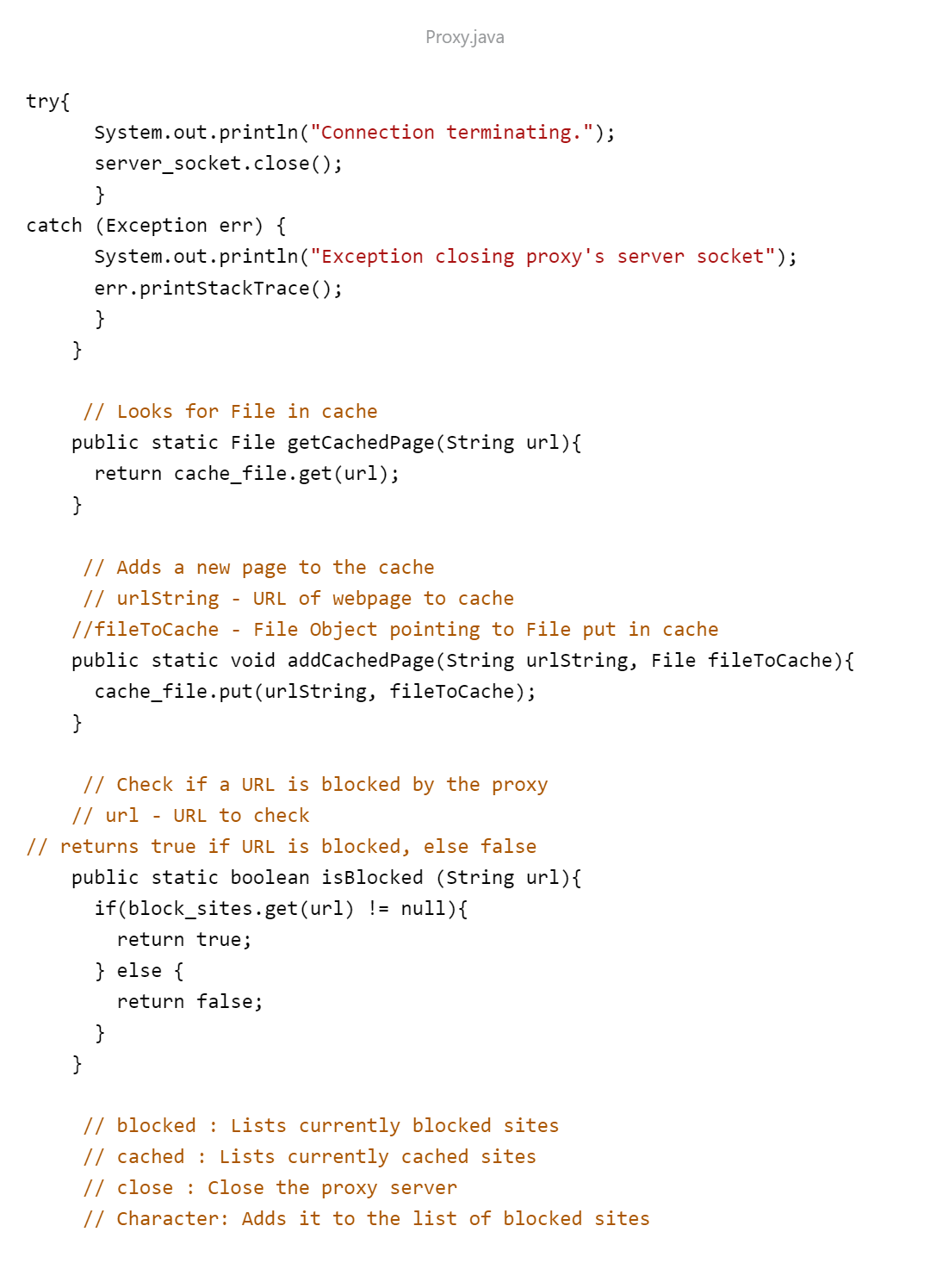
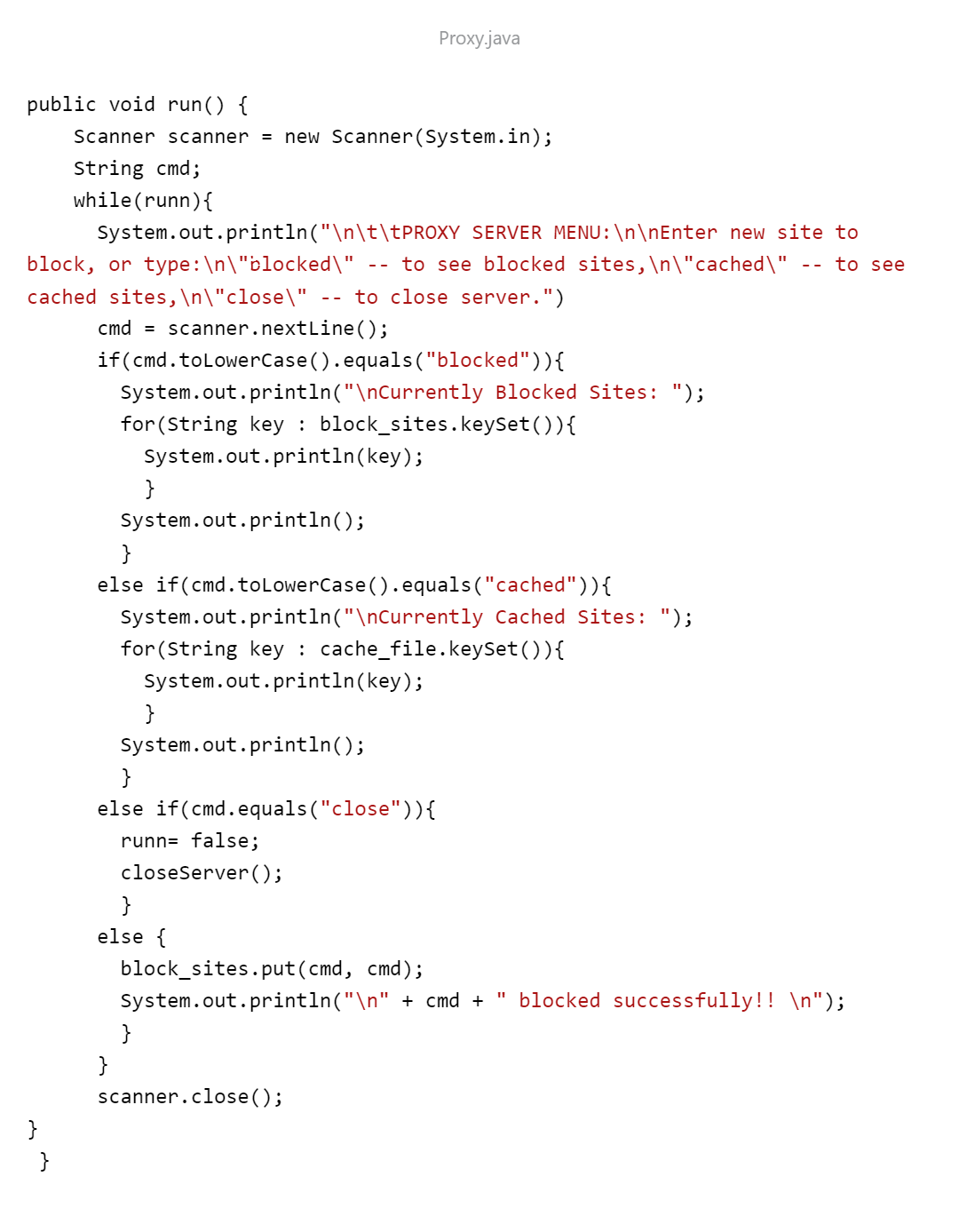
**SOURCE CODE**

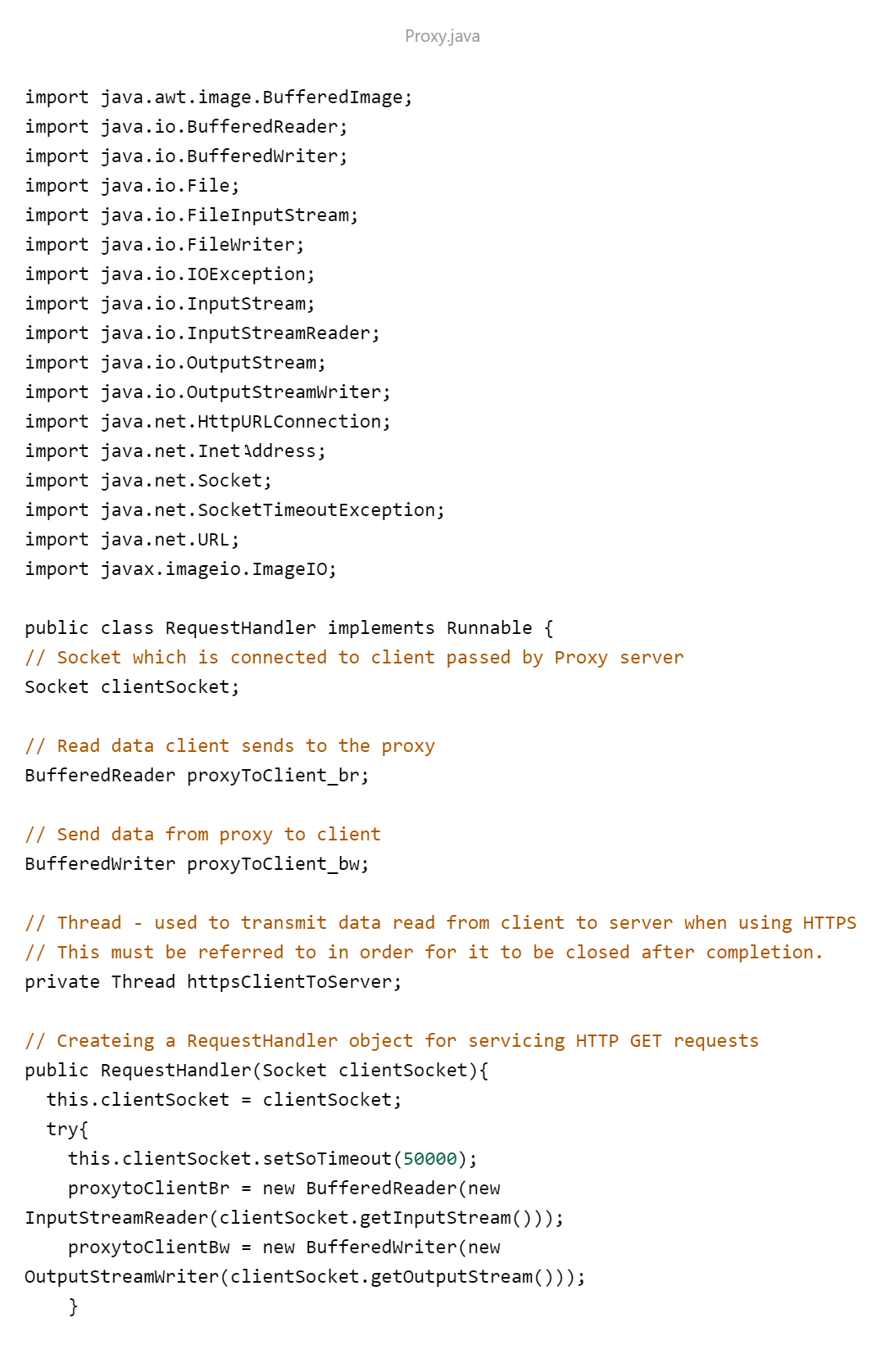
**Proxy java class :** 

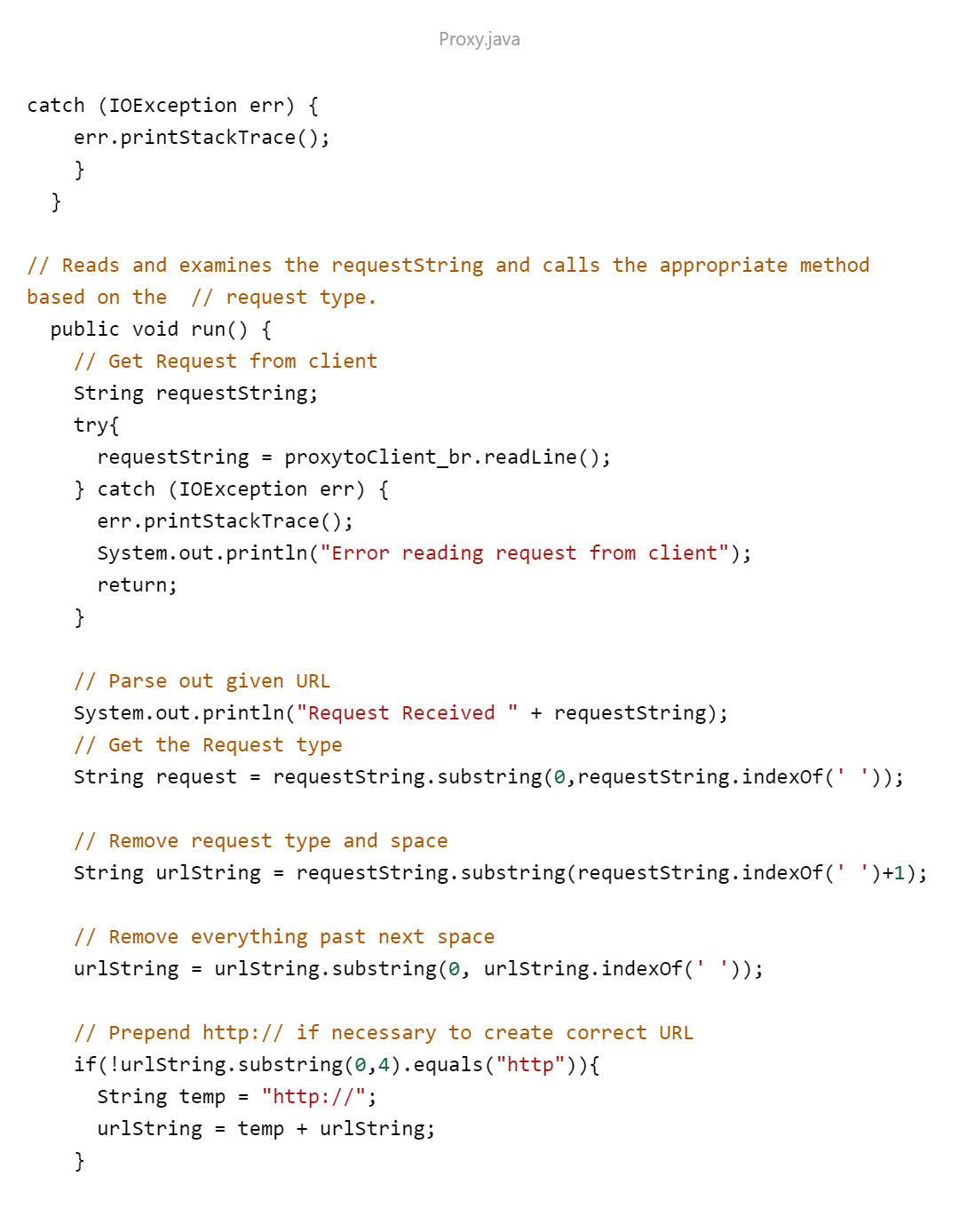
 



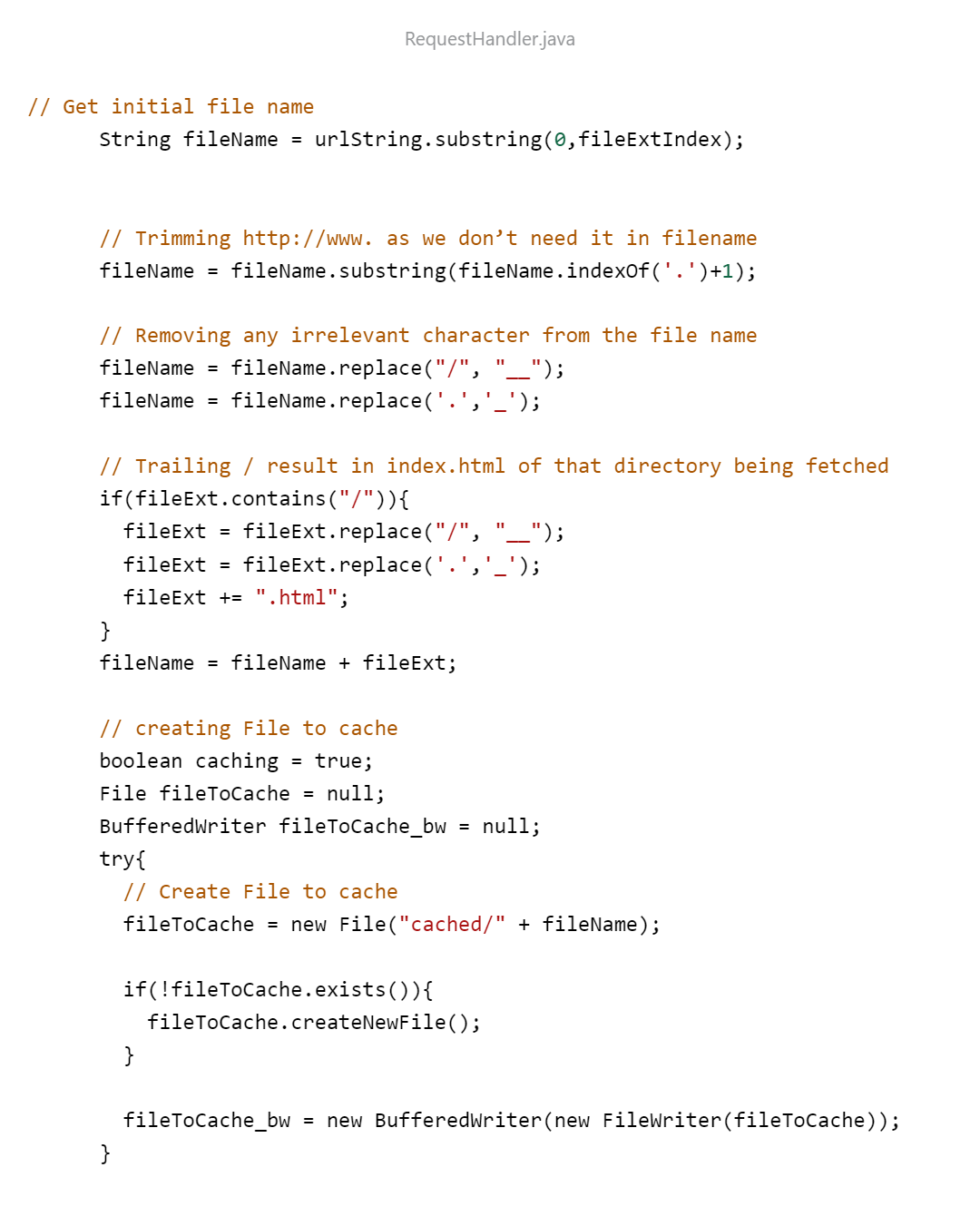
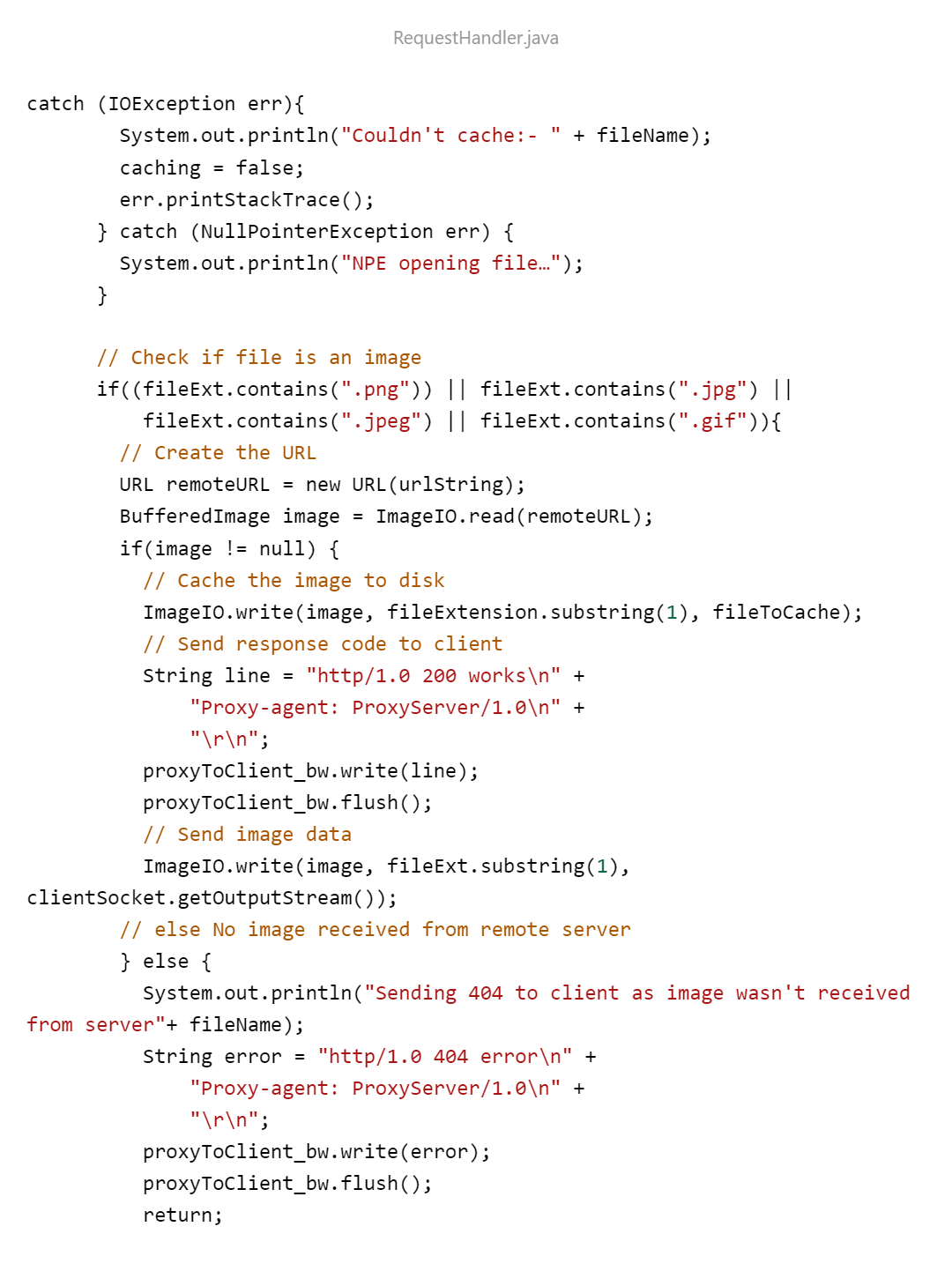
**RequestHandler class java:** 



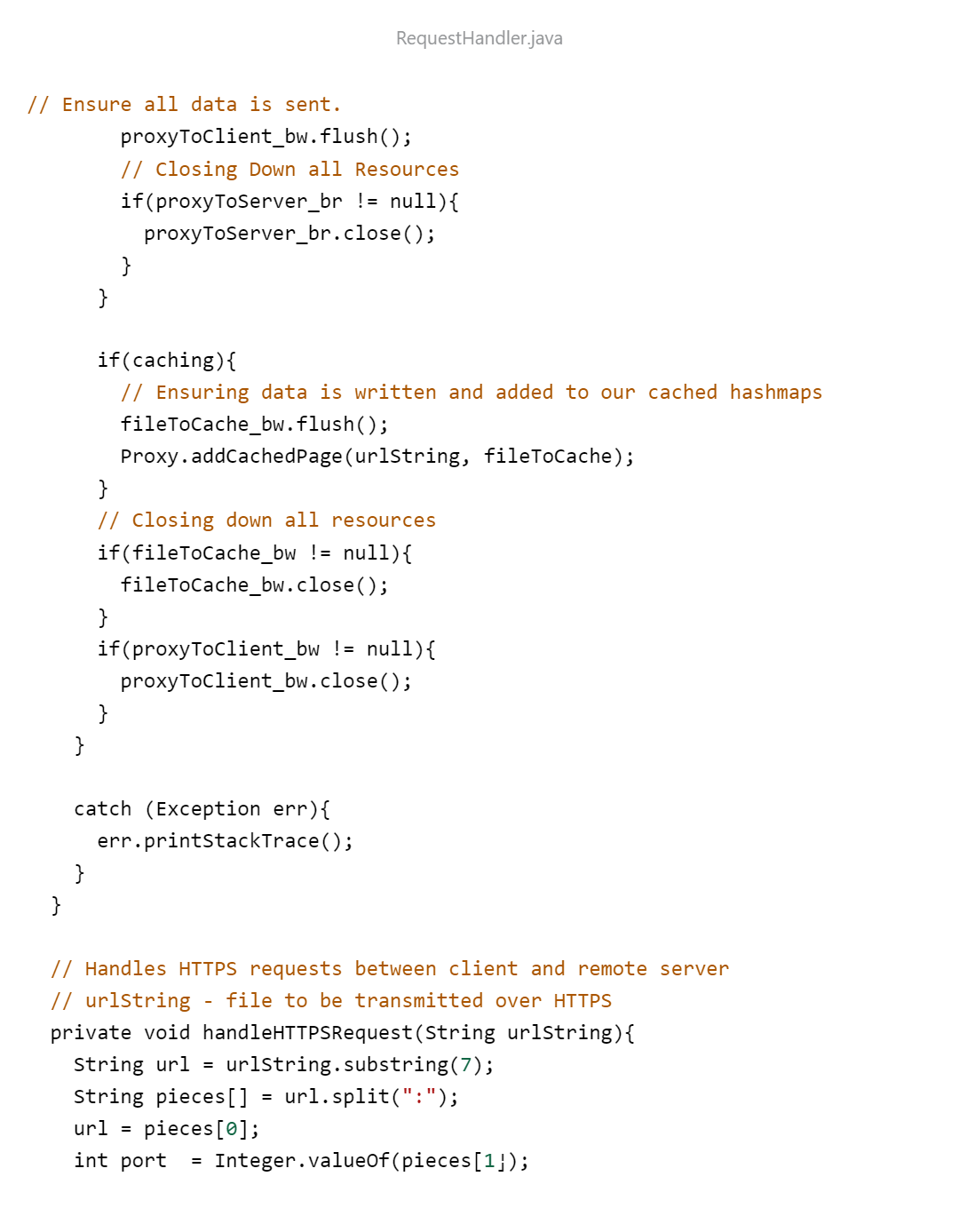
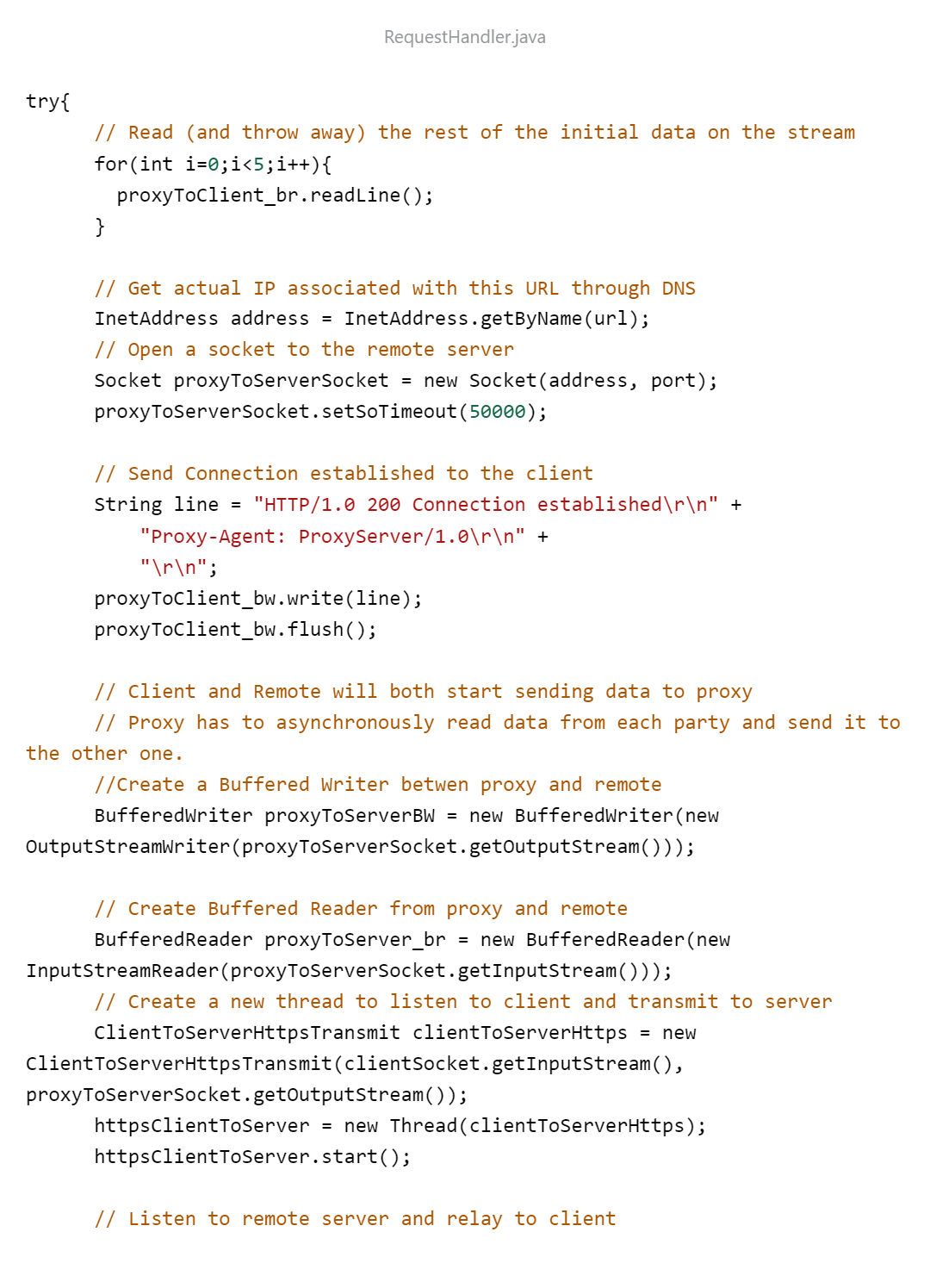
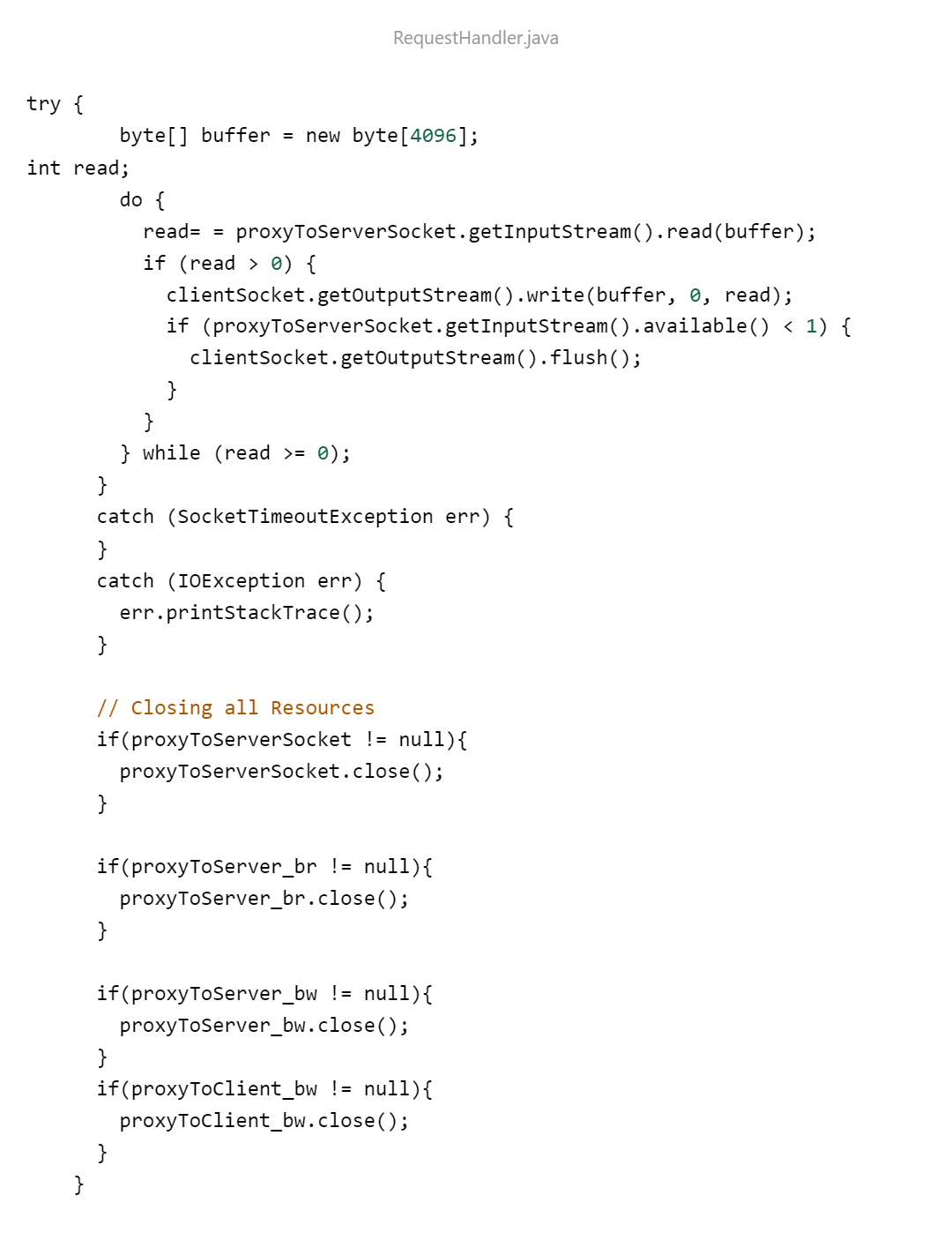
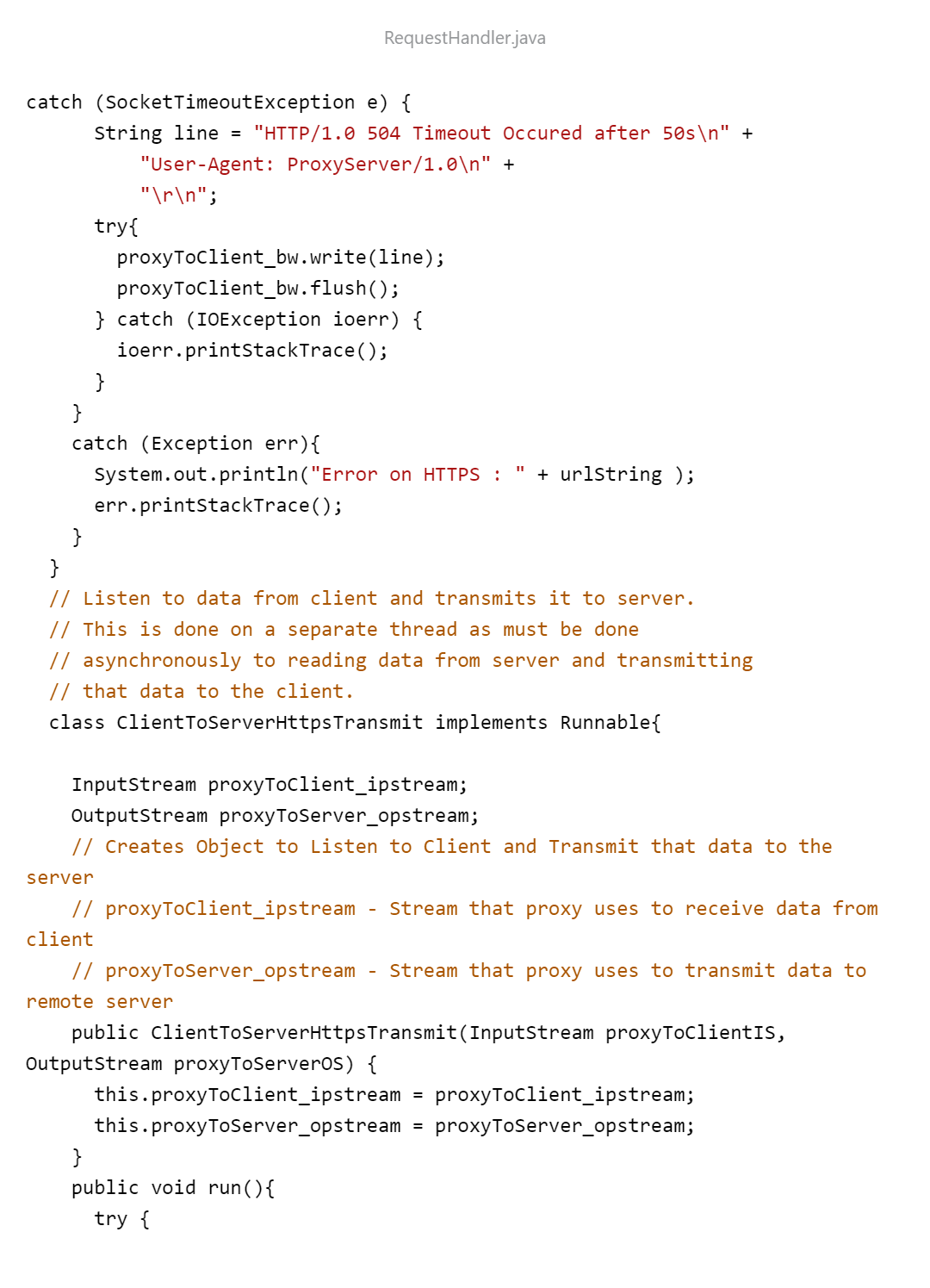
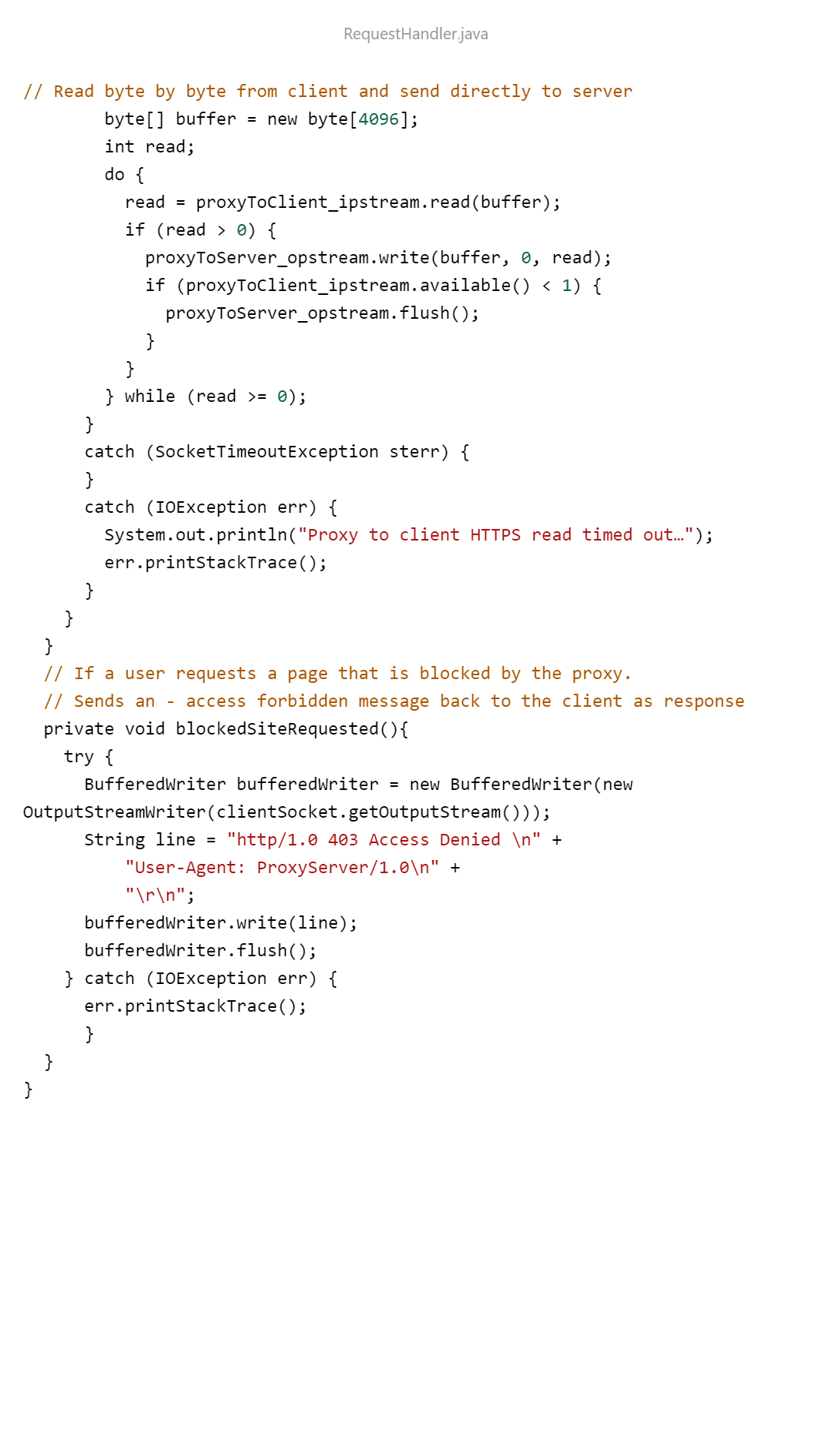








**SNAPSHOTS**

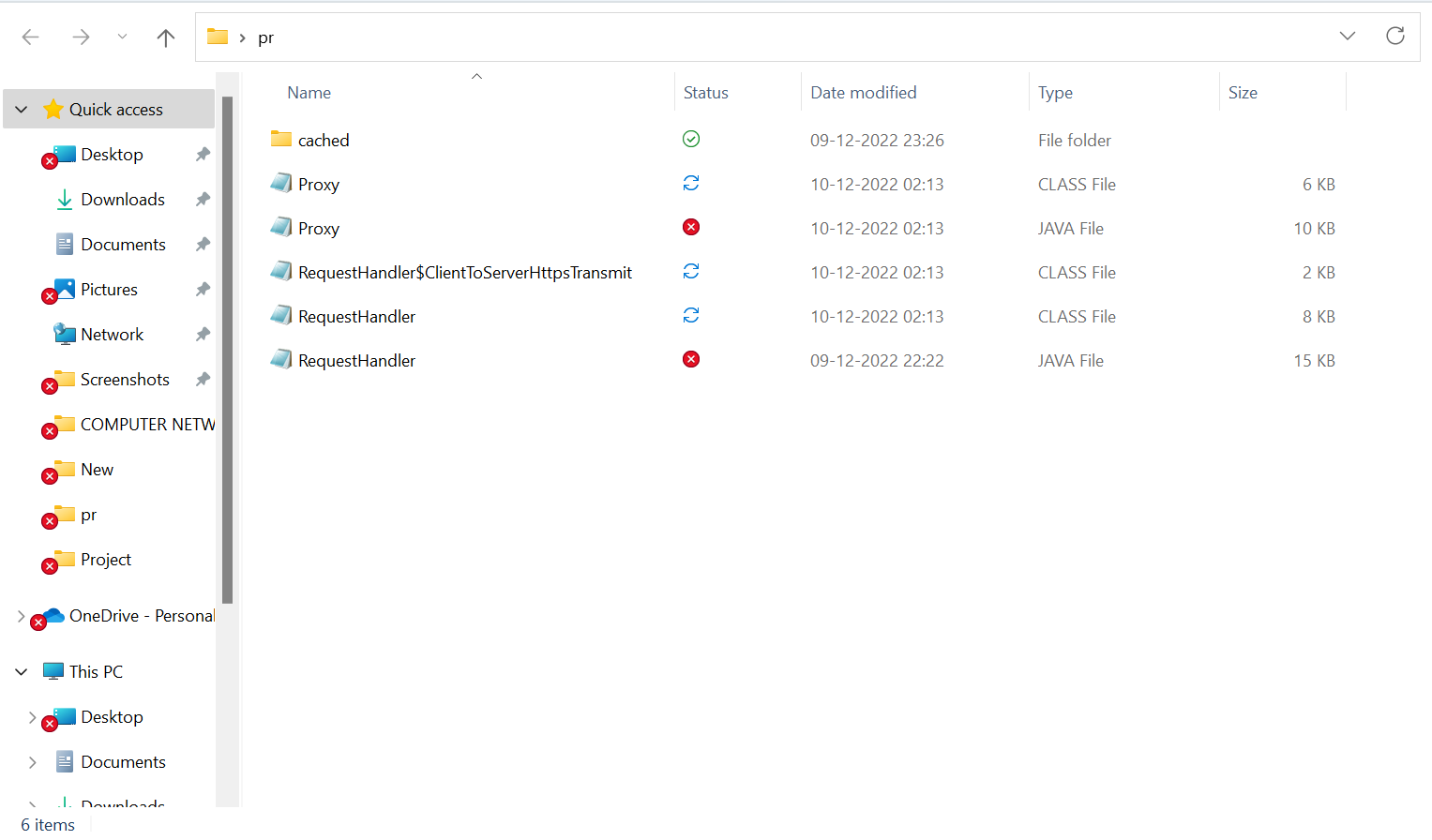
****

Fig. 3.1. Directory Consisting of required files and “cached” folder for storing and retrieving Cache Files



Fig. 3.2. Firefox setup to forward all traffics to port number 6969

* Creates cached\_sites.txt and block\_sites.txt as it’s not found in the directory

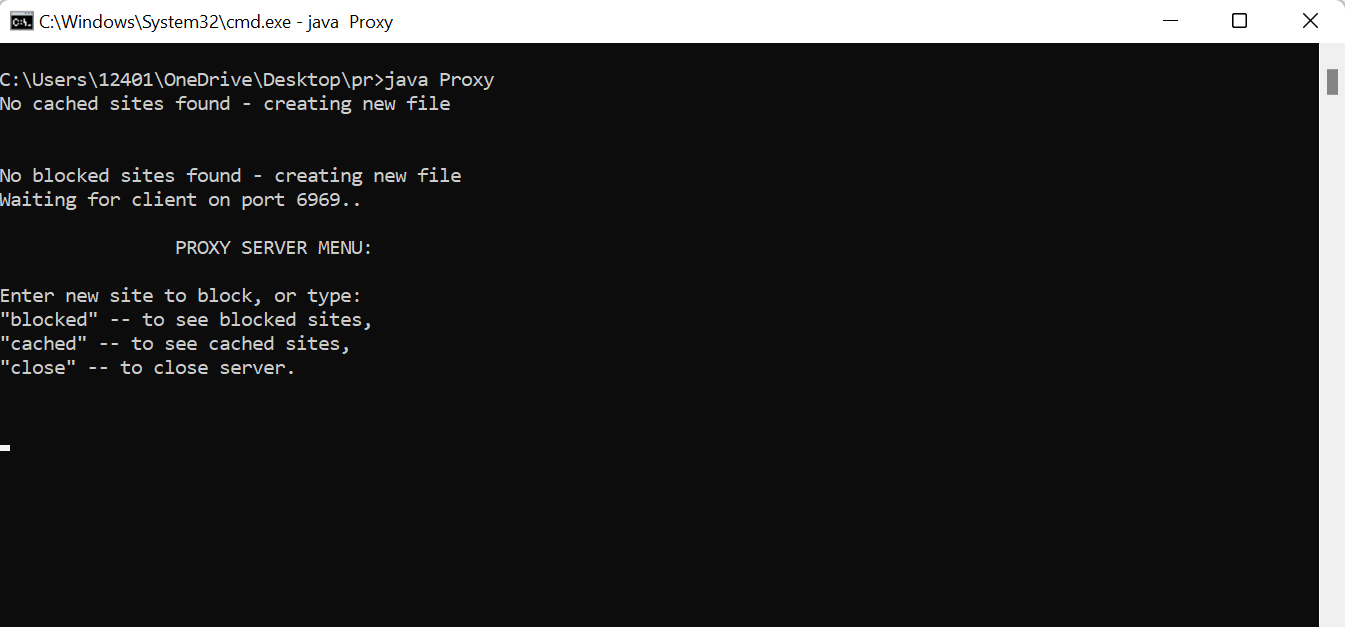


Fig. 3.3. Proxy Server Menu

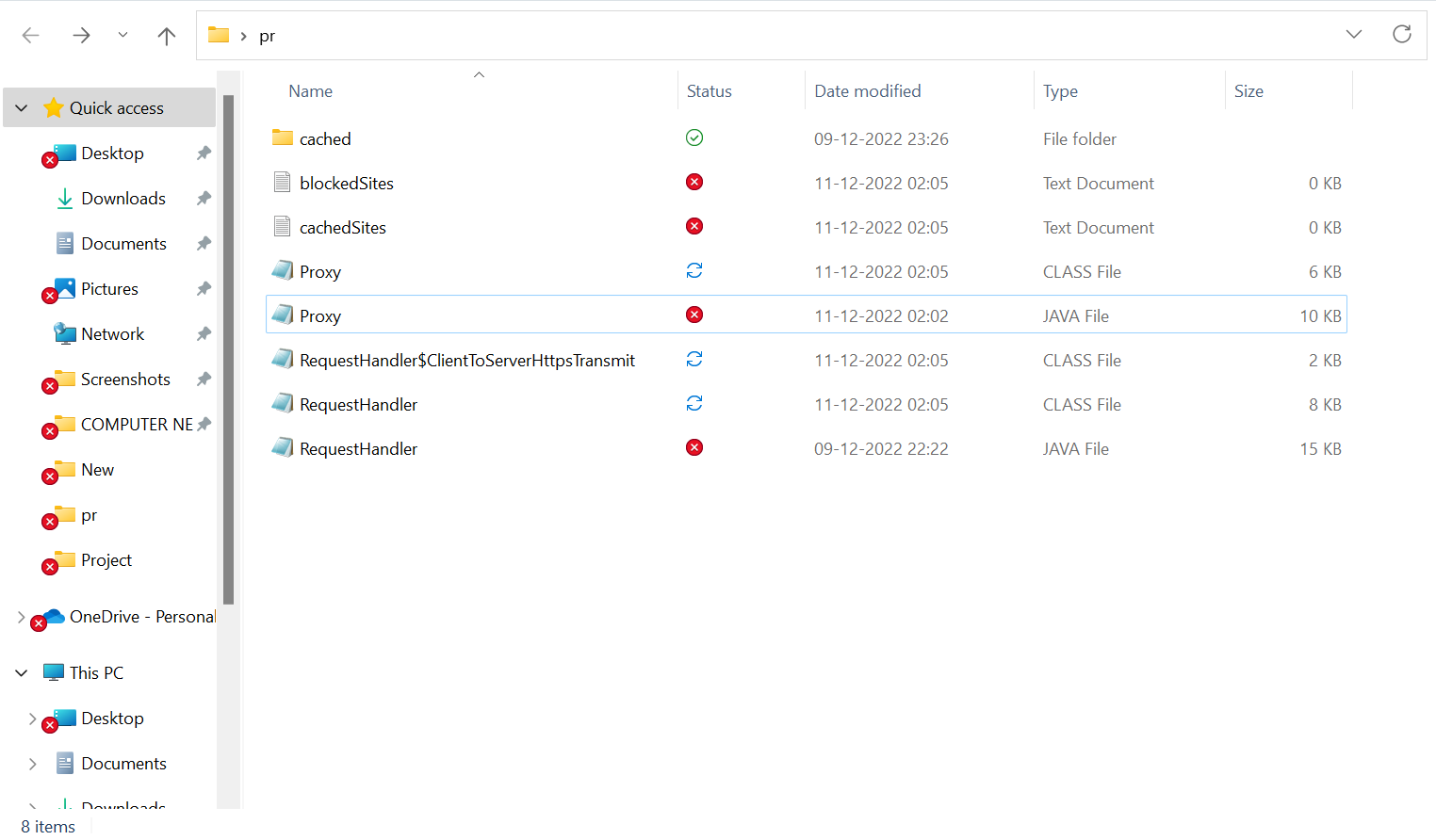


Fig. 3.4. Created Files in directory

**3.1. Caching:**

* The HTTP website “http://goodglowingquietrainbow.neverssl.com/online” is accessed to illustrate the working of HTTP GET Request and Response by remote server through Proxy server.
* When visited for the first time, the website is cached and stored in the local directory “cached” folder, and the key to access its cache is written in cached\_sites.txt.
* If visited anytime again, the Proxy server looks for a key in the text file and if found then locates the cache from the “cached” folder and returns the response to the client, avoiding connection with the remote server ( increasing response time).
* Input “cached” is given to view cached sites.

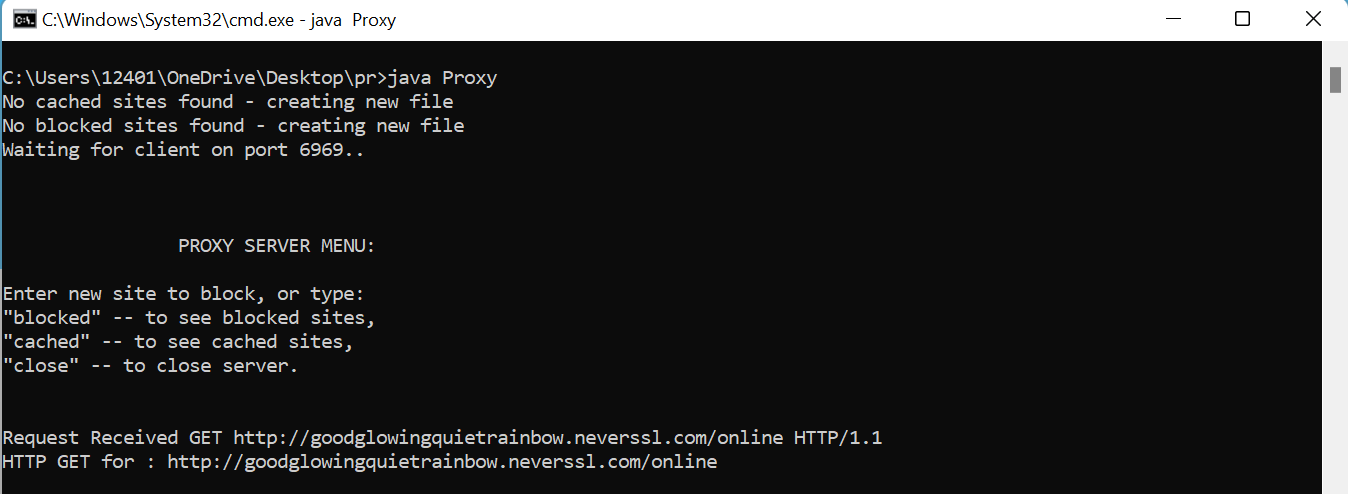


Fig. 3.5. HTTP GET Request

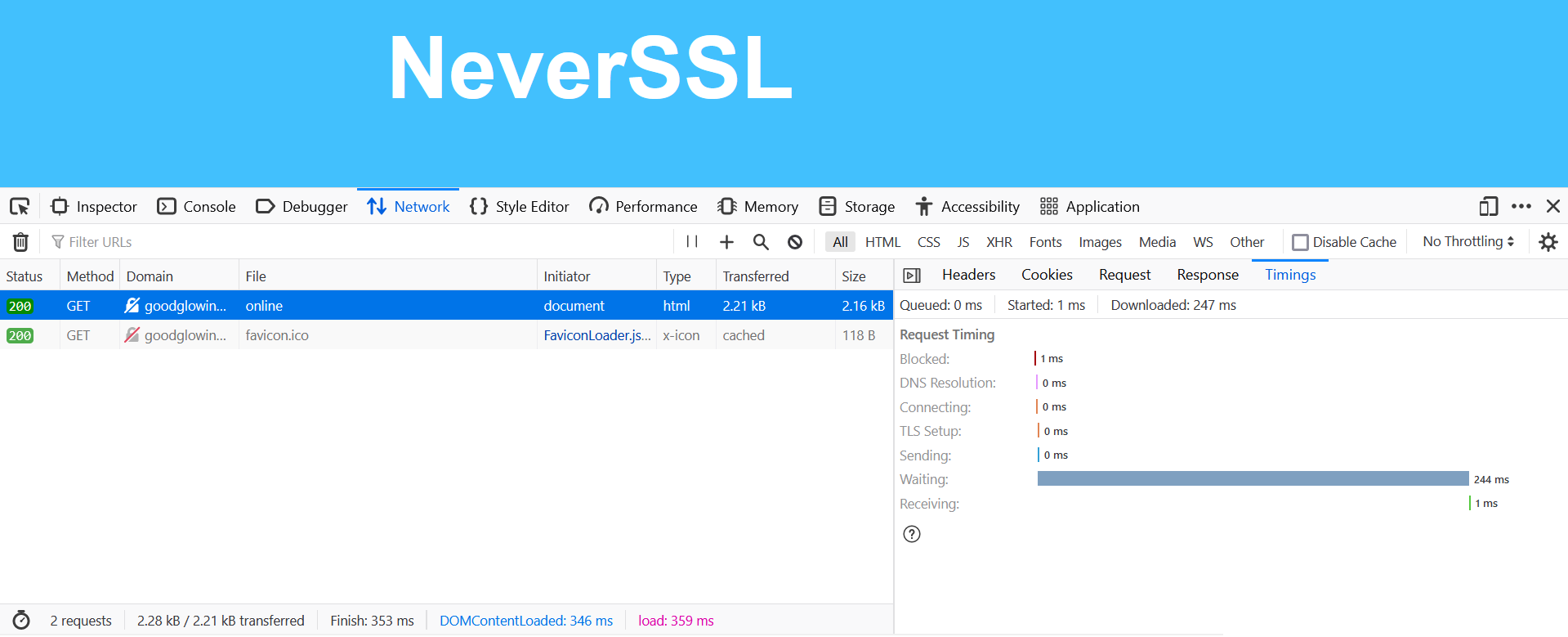


Fig. 3.6. Inspected website without caching (accessing first time)

Waiting time: 244 ms

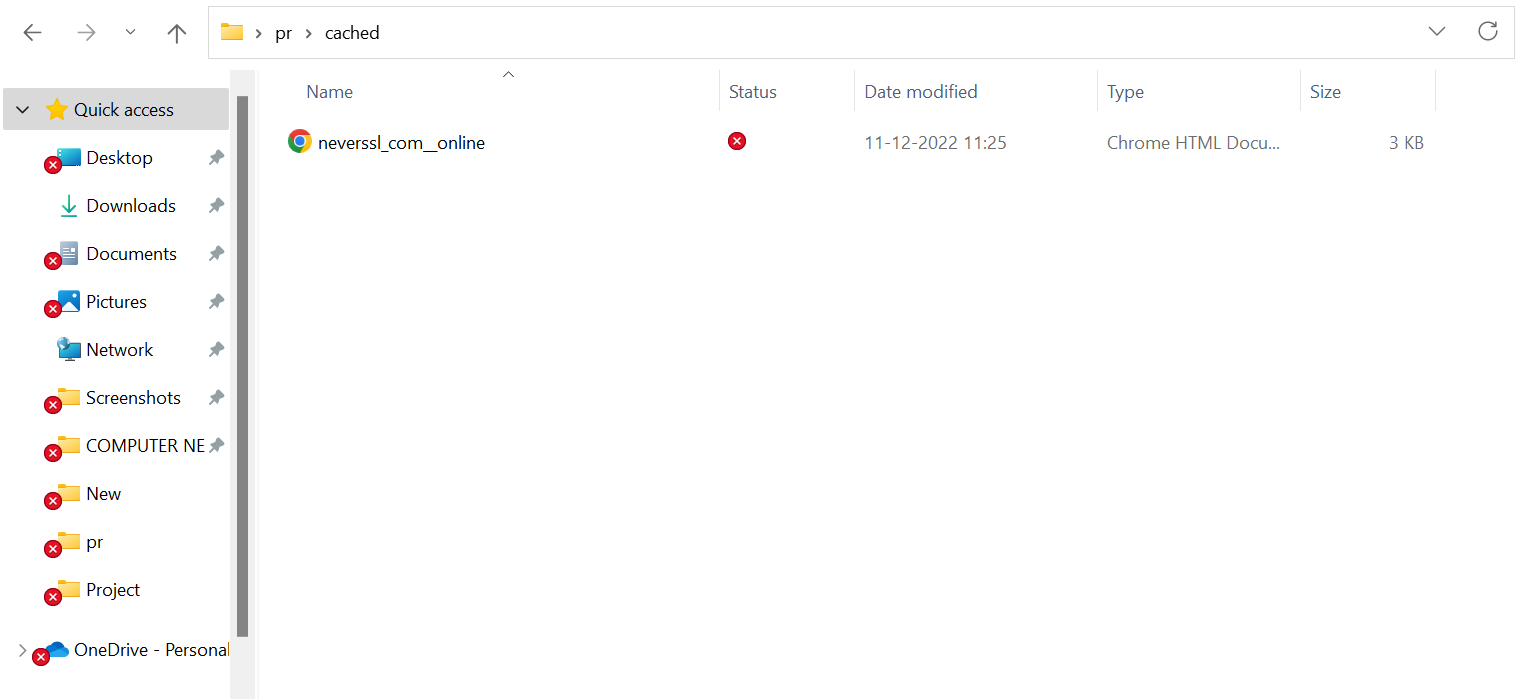


Fig. 3.7. Cached site in local folder

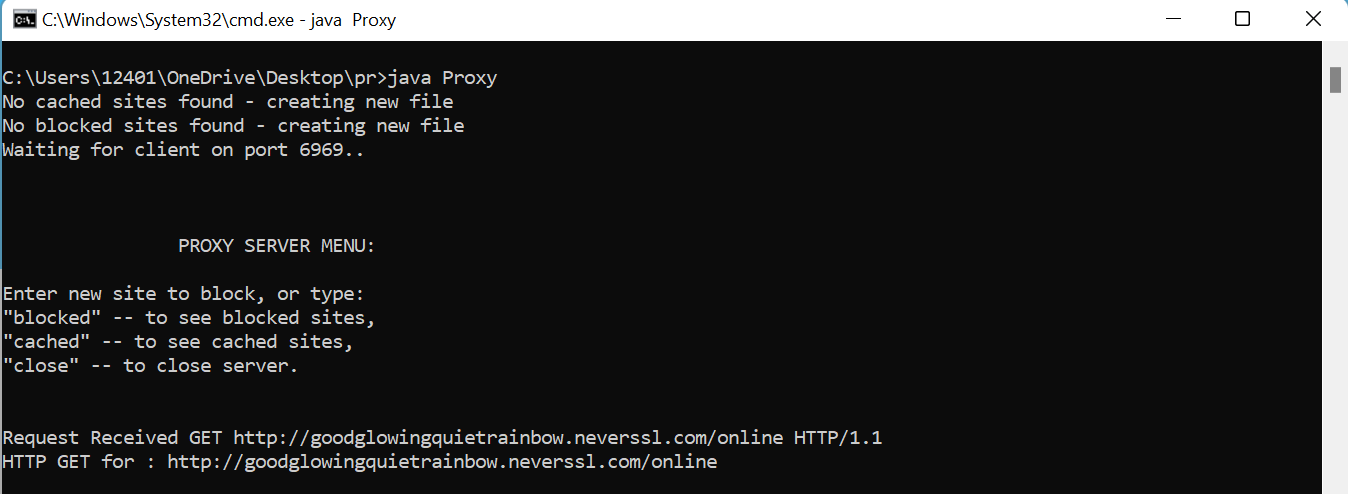


Fig. 3.8. Cached copy found in directory and given to client

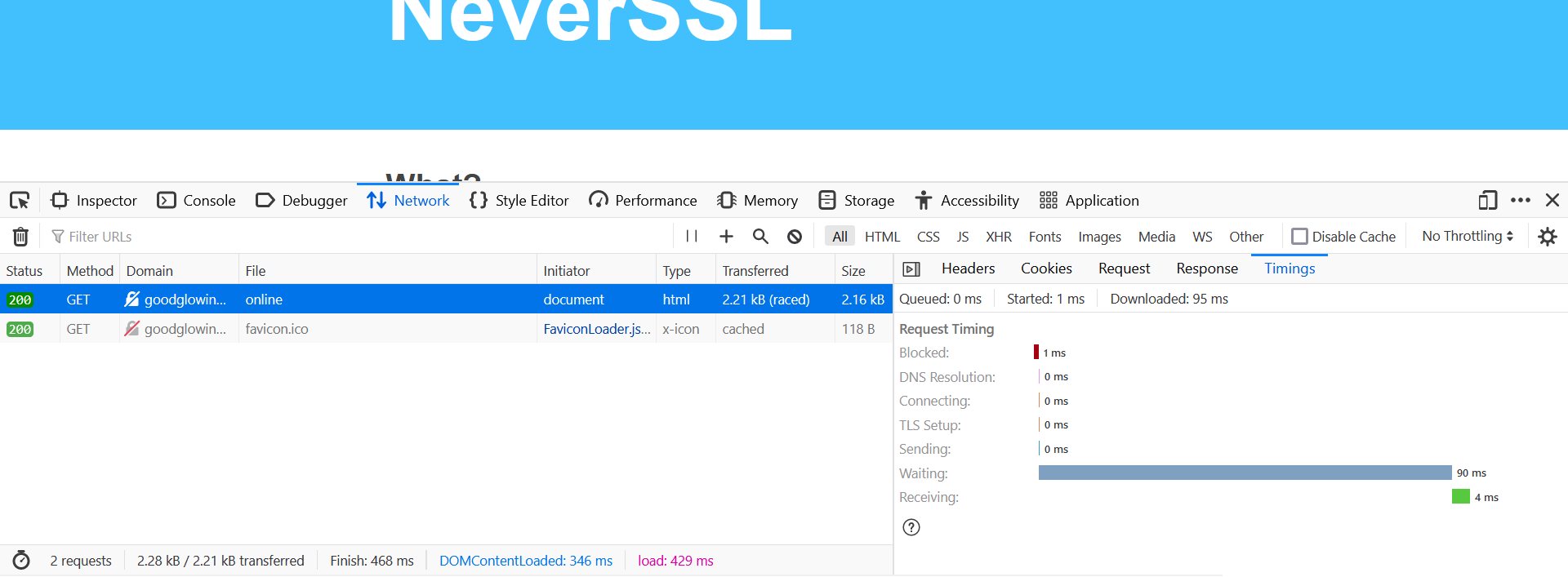


Fig. 3.9. Inspected website after caching

Waiting Time: 90 ms (retrieved from cache)

**3.2. Blocking:**

* “<http://goodglowingquietrainbow.neverssl.com/online>” is entered as input dynamically to block.
* “blocked” is given as input dynamically to show currently blocked sites.



Fig. 3.10. Request is blocked and list of blocked sites

* When requesting for “<http://goodglowingquietrainbow.neverssl.com/online>” , access is denied.

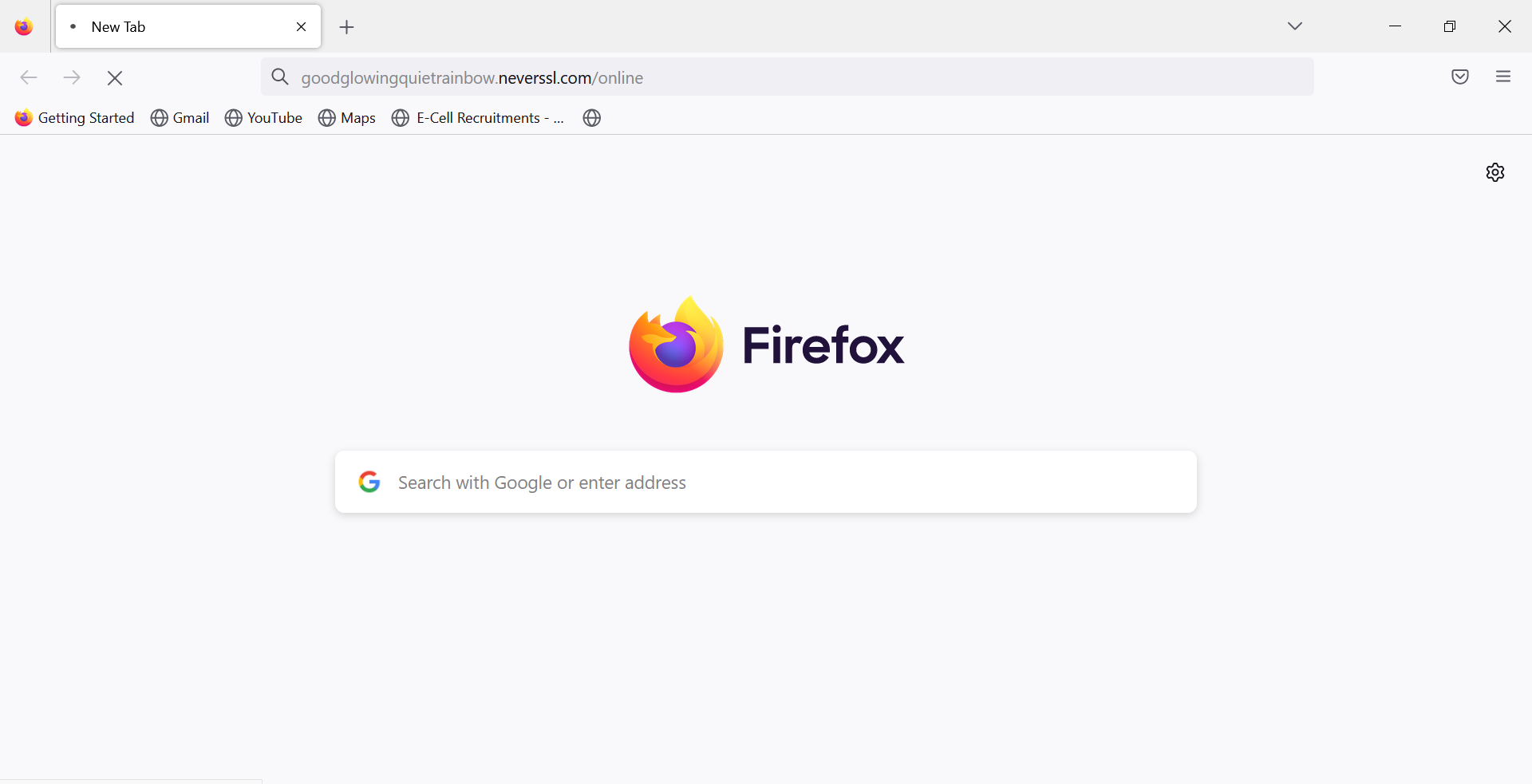


Fig. 3.11. Webpage is unable to load due to forbidden access



Fig. 3.12. Closing Proxy Server

**CONCLUSION AND FUTURE PLANS**

**4.1 CONCLUSION:**

The implemented HTTP Web Proxy Server can thus accept all HTTP GET requests and perform tasks like Caching and Blocking of the sites provided by the user. Caching increases network efficiency and reduces response time of sites. In case of a HTTPS request, the Proxy can accept the requests and create a secure channel between the client and the server but cannot interfere with the data transmitted between the two and hence cannot cache or block HTTPS sites. However, the encrypted data (using SSL) from the client is transferred to the server through the intermediary Proxy and no third party can have access to the data since the key to decrypt the encrypted data is present only with the client and the remote server

**4.2 FUTURE PLANS:**

1. In order to improvise user interaction, I plan to further develop this project into a GUI based application where features like threads occupied/available, individual request types and requesting sites can be displayed. These features are only currently available through the command prompt. Other features, such as performance analysis and response time, might be easier to check in one location rather than individually in the browser.
2. We are unable to cache HTTPS connections for security reasons. But if we can figure out a means to overcome this, we can drastically cut down on response time. I further intend to research the issues surrounding the one mentioned in order to find a way to implement cache without leaving any security holes exposed.

**REFERENCES**

James F. Kurose, Amherst Keith Ross (2021), *Computer Networking: A Top Down Approach 8th Edition*. [*https://gaia.cs.umass.edu/kurose\_ross/online\_lectures.htm*](https://gaia.cs.umass.edu/kurose_ross/online_lectures.htm)

Martin Sysel, Ondřej Doležal (2014), Research paper on *An Educational HTTP Proxy Server*

What is a Proxy Server?, [*https://www.fortinet.com/resources/cyberglossary/proxy-server*](https://www.fortinet.com/resources/cyberglossary/proxy-server)

Handling requests- [*https://www.baeldung.com/java-http-request*](https://www.baeldung.com/java-http-request)

Required Java Libraries for Proxy, [*https://docs.oracle.com/javase/8/docs/technotes/guides/net/proxies.html*](https://docs.oracle.com/javase/8/docs/technotes/guides/net/proxies.html)

Configure proxy settings, *https://www.java.com/en/download/help/proxy\_setup.htm*