**Step 1: Proxy Opens a Socket and Listens for Client Requests**

**🔹 What Happens?**

* The **proxy must create a TCP socket** and **bind it to an IP (127.0.0.1) and a port (8001)**.
* The **proxy listens** for incoming client requests (e.g., from a browser).
* The **proxy accepts the connection** and **logs the client's request**.

**🔹 Hints for Implementation**

A screenshot of a computer code

Description automatically generated

**Example: Browser Requests a Web Page via Proxy**

**Scenario:**

A **user opens their browser** and **enters the following URL**

[**http://zebroid.ida.liu.se/fakenews/test2.txt**](http://zebroid.ida.liu.se/fakenews/test2.txt)



A screenshot of a computer

Description automatically generated

**Step 2: Proxy Receives the Raw HTTP Request**

**🔹 What Happens?**

* The **browser sends an HTTP GET request** to the proxy instead of the web server.
* The **proxy reads the request** as **raw bytes** from the client connection.
* The **proxy prints/logs the request** for debugging.

A screenshot of a computer program

Description automatically generated

**Example: Proxy Receives and Logs Raw Request**

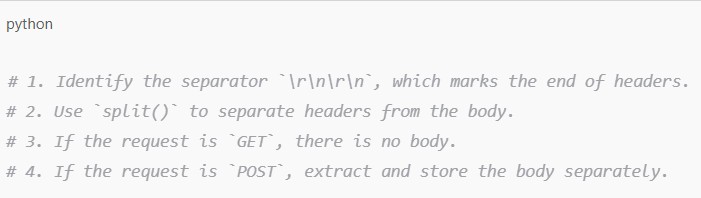
**b'GET /fakenews/test2.txt HTTP/1.1\r\nHost: zebroid.ida.liu.se\r\nUser-Agent: Mozilla/5.0\r\nAccept: text/plain\r\nConnection: keep-alive\r\n\r\n'**

The above get request is sent to the proxy

**Step 3: Proxy Splits Headers from the Body**

**🔹 What Happens?**

* The **proxy must separate the HTTP headers from the body** (if present).
* The **headers contain metadata** (like Host, User-Agent).
* The **body contains data** (only for POST requests, not GET).



**Example 1: GET Request (No Body)**

**Raw Request Sent by Browser**

A computer screen shot of text

Description automatically generated

**Extracted Headers**

A screenshot of a computer code

Description automatically generated

**Extracted Body**



**Step 4: Proxy Parses the Headers Into a Dictionary**

**🔹 What Happens?**

* The proxy has extracted the raw headers, but they are in plain text.
* The proxy must convert them into a dictionary for easy access.
* The first line (GET /path HTTP/1.1) is special and must be handled separately.
* The remaining lines contain key-value pairs (Header-Name: Value).

**🔹 Hints for Implementation**

A white screen with black text

Description automatically generated

**Example 1: Parsing Headers From a GET Request**

**Raw Headers Received by Proxy**

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Step 5: Proxy Resolves the Web Server’s IP Address**

**🔹 What Happens?**

* The client provided a hostname (zebroid.ida.liu.se) in the Host header.
* The proxy needs to find the actual IP address of the hostname.
* It does this by using DNS resolution (socket.gethostbyname()).
* The proxy then logs the resolved IP address for debugging.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

**Step 6: Proxy Forwards the Request to the Web Server**

**🔹 What Happens?**

* The proxy has the client’s full request and the server’s IP address.
* It must open a new connection to the web server (130.236.218.50).
* It sends the original request to the web server exactly as received.
* The proxy must wait for the server’s response after sending the request.

A screenshot of a computer

Description automatically generated

**Step 7: Proxy Receives the Response From the Web Server**

🔹 **What Happens?**

* The web server processes the request and sends a response back to the proxy.
* Since text files and images can be large, the server sends data in chunks instead of all at once.
* The proxy must read the data in chunks until the full response is received.
* The proxy prints/logs the received response for debugging.

A screenshot of a computer program

Description automatically generated

**Step 8: Proxy Modifies the Response (Optional)**

**🔹 What Happens?**

* The proxy can modify the response before sending it to the client.
* A common example is replacing specific words in text responses.
* The proxy must update the Content-Length header if the modification changes the size of the data.

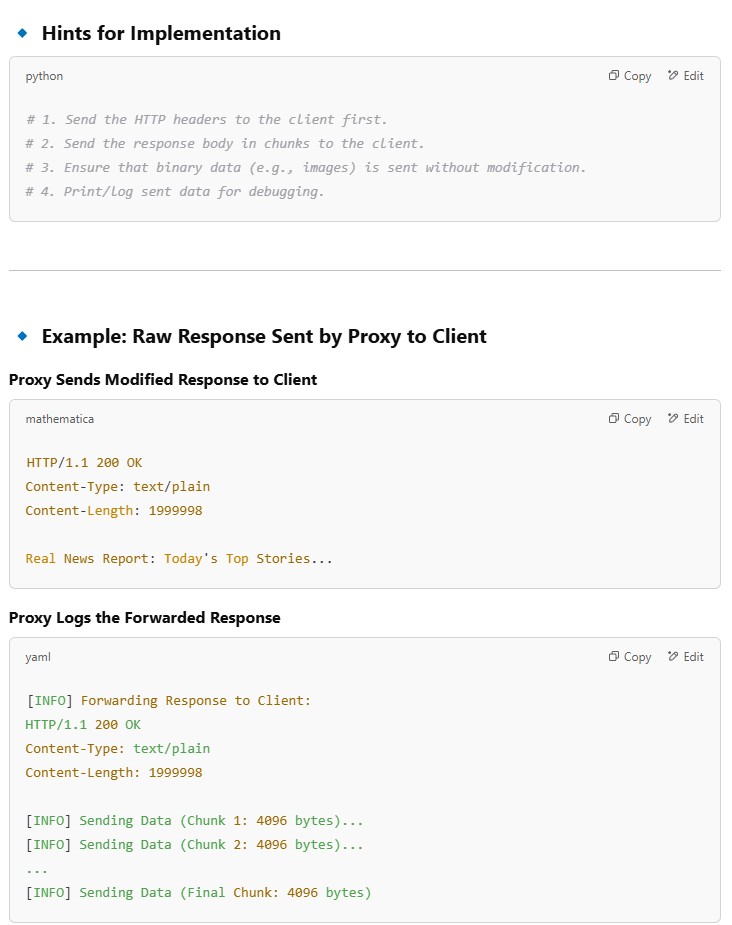
A screenshot of a web page

Description automatically generated

**Step 9: Proxy Sends the Response to the Client**

**🔹 What Happens?**

* The proxy must send each chunk immediately to the client (browser).
* It ensures that the browser receives the data in proper HTTP format.
* The browser automatically processes and renders the content once fully received.



**Step 10: How the Browser Processes the Received Data**

**🔹 What Happens?**

* The proxy has sent the full response to the client (browser).
* The browser reads the HTTP response and checks the Content-Type header.
* If the Content-Type is text-based, the browser renders it as a web page or plain text.
* If the Content-Type is image-based, the browser renders it as an image.

A screenshot of a computer

Description automatically generated

**Why Images Require Special Handling?**

**🔹** Text Responses (Content-Type: text/plain, text/html)

* Can be decoded and modified before being sent to the client.
* Example: Replacing "Fake News" with "Real News" in a text response.

🔹 Image Responses (Content-Type: image/jpeg, image/png)

* Contain binary data (not human-readable text).
* Cannot be decoded as text without corrupting the image.
* Must be forwarded exactly as received, without modification.

**Step 11: Handling Image Files in a Proxy**

**🔹 What Happens?**

* The client requests an image (GET /test5.jpg), and the proxy forwards the request.
* The web server responds with binary image data (Content-Type: image/jpeg).
* The proxy reads the image data in chunks (to handle large images).
* The proxy forwards the binary data exactly as received, without modification.
* The browser receives the image and renders it correctly.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

**Handling Large Image Files Efficiently**

**🔹 Why Chunked Transfer is Necessary?**

* Image files can be large (e.g., several MBs).
* Reading the entire image at once can cause memory overload.
* Instead, the proxy should read and forward the image in small chunks.

A screenshot of a computer program

Description automatically generated