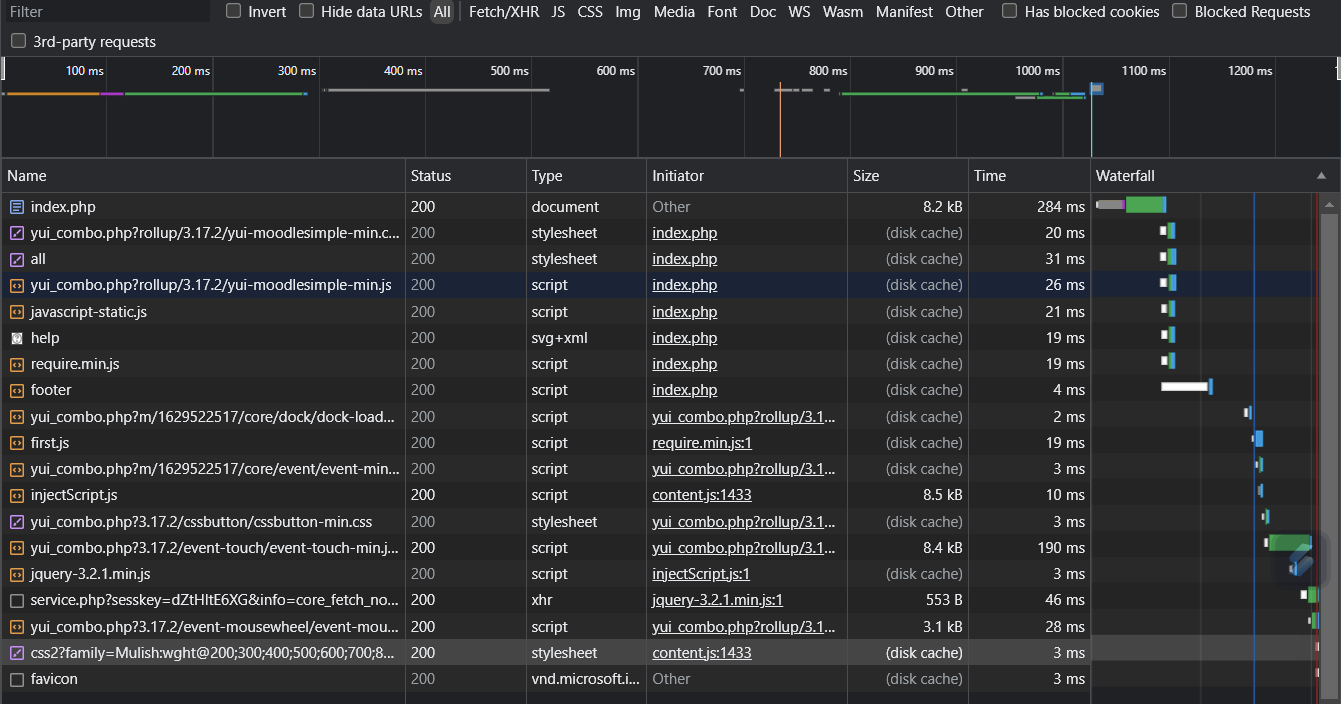
MOODLE REPPORT:

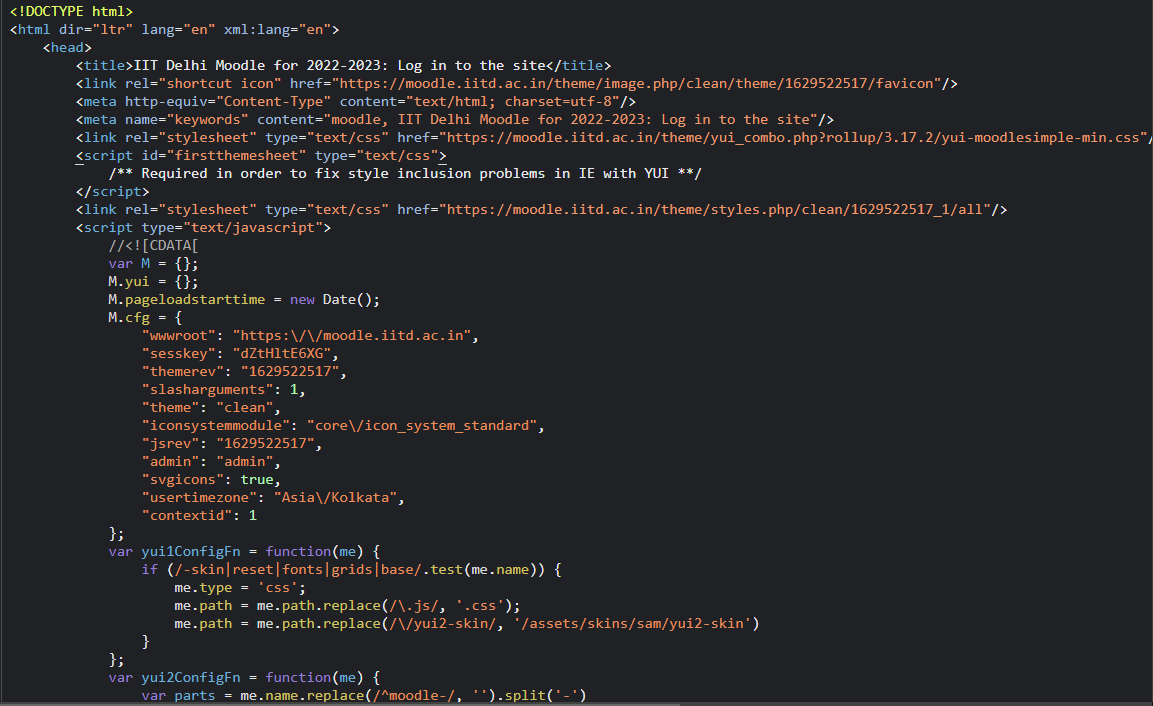
**1. The webpage** looks as above:

And the network console is like :



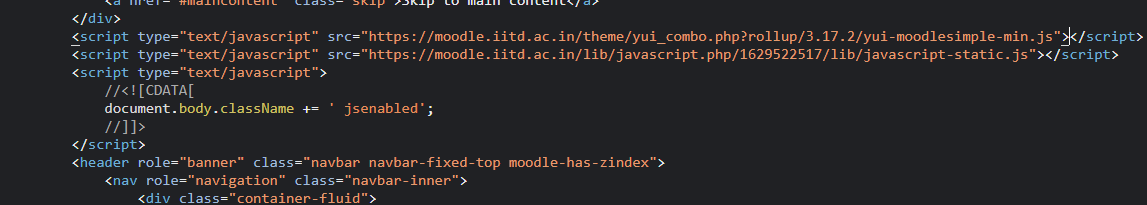
The browser requests many different files like php, stylesheets, scripts, svgs and xmls.

The requests are mainly initiated in index.php file:



But after that the requests are also initiated in other files like script and all…

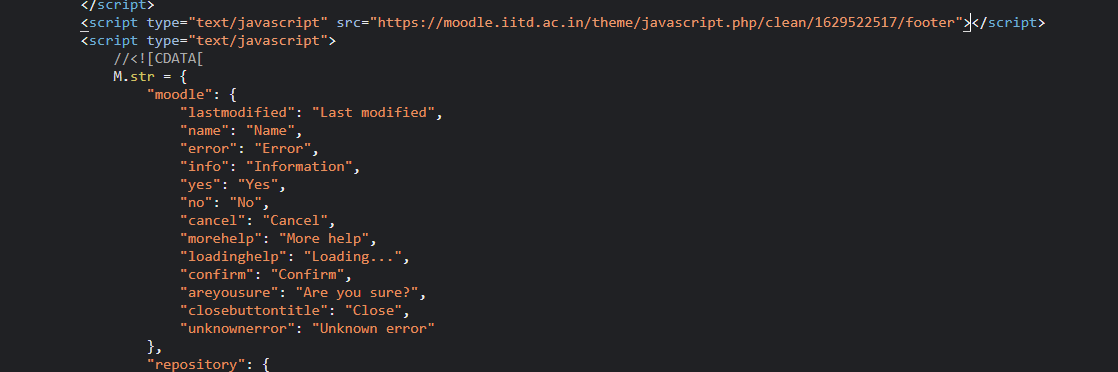
Here we can see initiation of css files named moodlesimple-min.css, firstthemesheet and clean/all



Initiation of javascript docs…



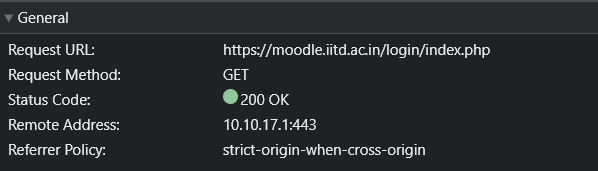
More script files:



Script for footer:

///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

**2. Headers:**



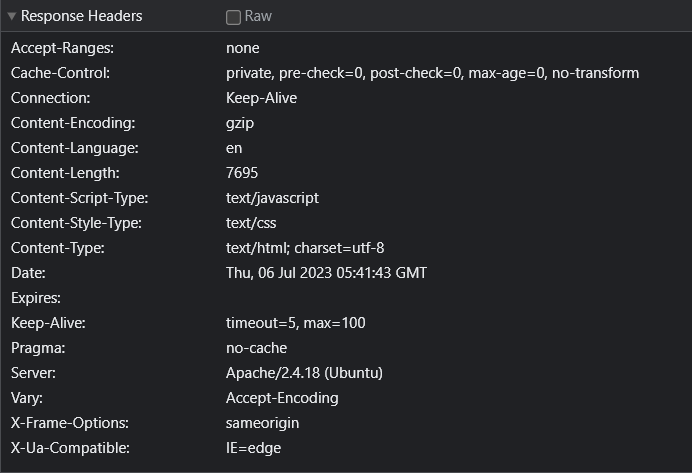
Status code refers to http response which is 200 showing successful response.

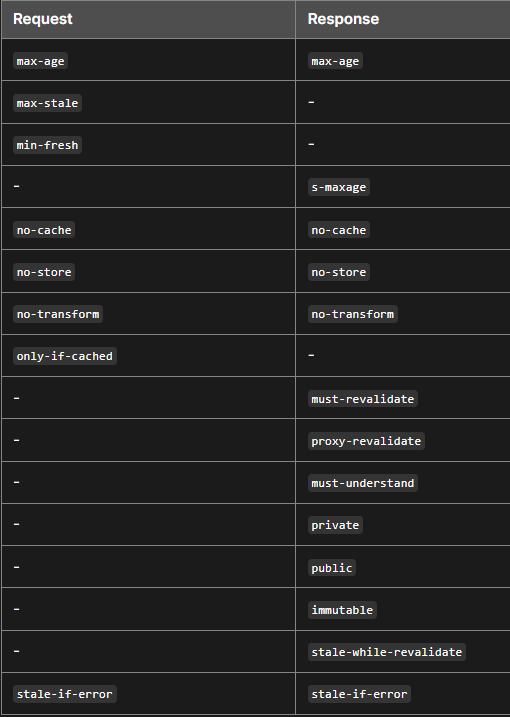
Remote address probably refers to the port

Referrer policy refers to whether url is sent full or only a part of it.

Strict-origin-when-cross-origin offers more privacy. With this policy, only the origin is sent in the Referer header of cross-origin requests. This prevents leaks of private data that may be accessible from other parts of the full URL such as the path and query string.

***Response headers*** hold additional information about the response, like its location or about the server providing it.



🡪The Cache-Control HTTP header field holds directives (instructions) — in both requests and responses — that control caching in browsers and shared caches (e.g. Proxies, CDNs). Private cache exists in the client. It is also called local cache or browser cache. It can store and reuse personalized content for a single user. The max-age=N response directive indicates that the response remains fresh until N seconds after the response is generated. Some intermediaries transform content for various reasons. For example, some convert images to reduce transfer size. In some cases, this is undesirable for the content provider. no-transform indicates that any intermediary (regardless of whether it implements a cache) shouldn't transform the response contents.

🡪Connection header is a hop by hop header and it should not be retransmitted by proxies or cached. The Keep-Alive general header allows the sender to hint about how the connection may be used to set a timeout and a maximum amount of requests.

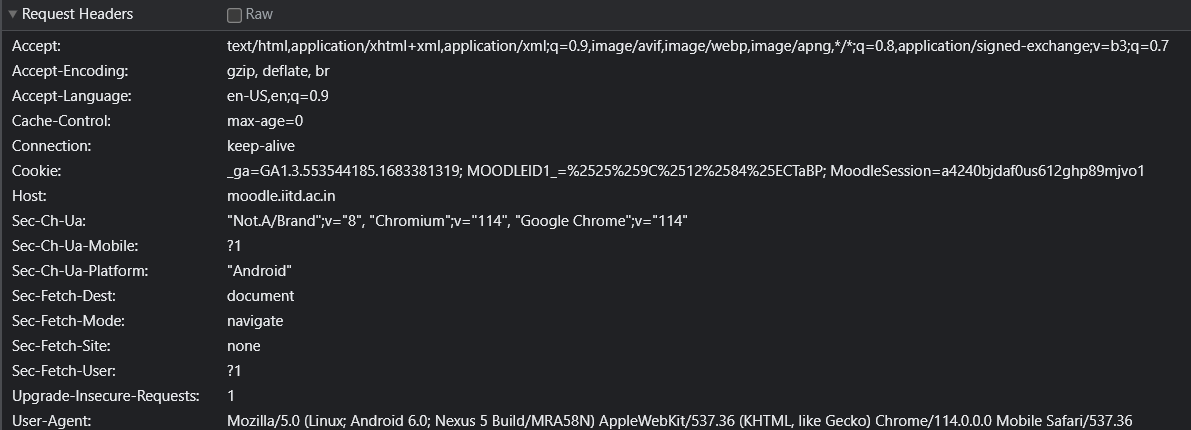
🡪Content- Used to specify the compression algorithm, human language, size of resource(in bytes), type of resource.

🡪Expires The date/time after which the response is considered stale.

🡪Pragma Implementation-specific header that may have various effects anywhere along the request-response chain. Used for backwards compatibility with HTTP/1.0 caches where the Cache-Control header is not yet present.

🡪Vary Determines how to match request headers to decide whether a cached response can be used rather than requesting a fresh one from the origin server.

***Request headers*** contain more information about the resource to be fetched, or about the client requesting the resource.



🡪Sec-Fetch metadata request headers provide information about the context from which the request originated. A server can use them to make decisions about whether a request should be allowed, based on where the request came from and how the resource will be used.

🡪Sec-Fetch-Site is set to none because the request has not been made by a site.

🡪Sec-Fetch-User is set to 1 indicating that a navigation request was triggered by user navigation.

🡪Sec-Fetch-Dest is set to socument indicating requests destination.

🡪Cookie Contains stored HTTP cookies previously sent by the server with the Set-Cookie header.

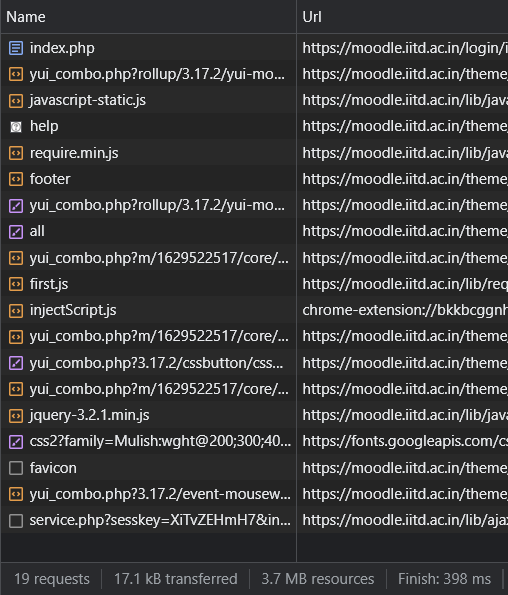
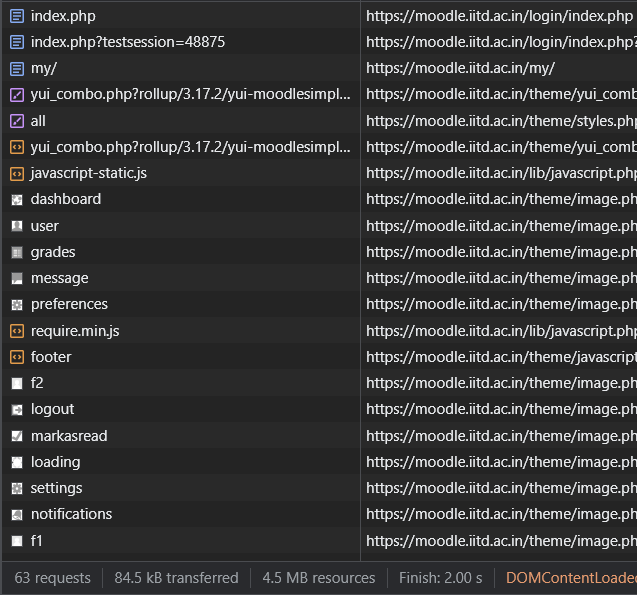
🡪User-Agent Contains a characteristic string that allows the network protocol peers to identify the application type, operating system, software vendor or software version of the requesting software user agent.

Representation or payload headers have not been used over here.

///////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////

**3. Login to moodle:**

Before: After:



After login to moodle the no. of requests went to 63 from 19 and it took 2s to finish.

When we login to moodle, a POST request is initiated by index.html to the server and the server responds by giving a get redirect request. After that a plethora of resources are demanded by the index.html and others which result in issuing of new get requests.

*Cookies* are small pieces of data that websites store on a user's web browser. They are a fundamental part of the communication between the web browser and the server and are often used for a variety of purposes, including authentication. Cookies allow a web server to remember and track information about a user's browsing activity, preferences, and state data.

This is how cookies are used in communication and authentication between the browser and the server:

Basic cookie mechanism:

When a user visits a website, the web server may send a small HTTP header named "Set-Cookie" to the user's browser. This header contains data (cookies) that the browser stores locally. The browser will then include the cookie in all future requests to the same website, allowing the server to identify and recognize the user.

Session cookies:

Session cookies are temporary cookies that are stored in the browser's memory and are typically used to store session information. When the user closes the browser, the session cookie is usually deleted and the session ends. They are typically used to authenticate users during a browsing session.

Persistent cookies:

Persistent cookies have a specified expiration date and are stored on the user's device even after the browser is closed. They are used for a variety of purposes, such as remembering login information, tracking user preferences, and providing a personalized browsing experience.

accuracy:

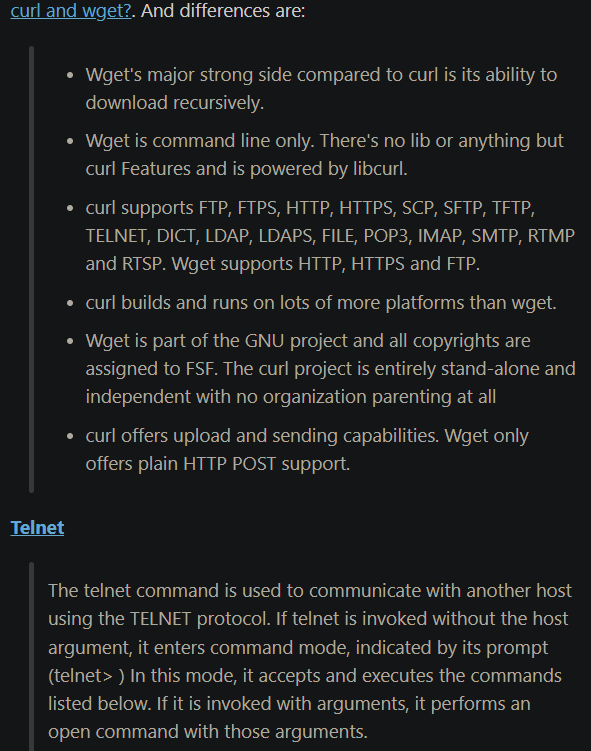
Cookies play an important role in user authentication. When a user logs into a website, the server generates a unique ID or token (usually a session ID) that represents the authenticated session. This identifier is then stored in a cookie on the user's browser. The browser includes this cookie in each subsequent request to the server, allowing the server to identify the user and keep their session authenticated. Security vanity:

Although cookies are widely used for authentication, they also raise security concerns. For example, if an attacker goes to a user's browser and extracts their authentication cookie, then they can impersonate the user without their credentials. To work around this, websites use various security measures such as HTTPS, secure flags on cookies, and token expiration to limit the risks associated with cookies.

Same-origin policy:

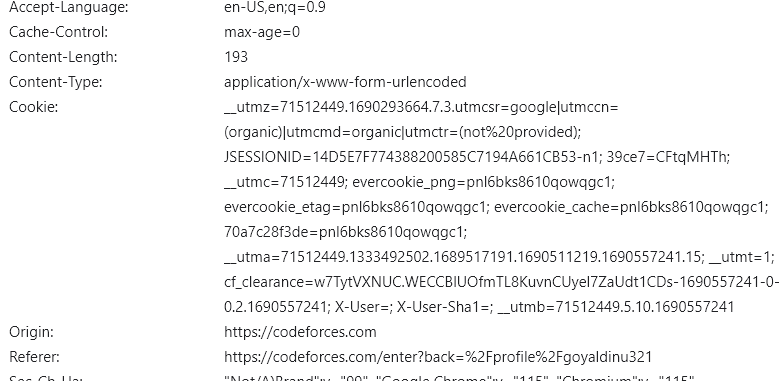
Cookies are subject to a "same-origin policy", which means that a website can only access cookies set by its own domain and not by other domains. This policy improves security by preventing different websites from accessing each other's data stored in cookies. In summary, cookies are essential for maintaining state information in browser-server communication and are widely used for user authentication, session management, and personalization on web pages. They enable a seamless and personalized user experience while requiring appropriate security measures to protect user data and privacy.

BONUS TASKS :



JSON Web Tokens (JWTs) are a type of token-based authentication that is commonly used in web applications. JWTs are compact and self-contained, making them an efficient way to transmit authentication information between parties. They consist of three parts separated by dots: the header, the payload, and the signature.

Codeforces as a website does not use jwts as can be seen in the screenshot attached as there is no sign of a authorization header or a long string of text in the cookies.



This is a