TASK -1

1. HTTP1.1 VS HTTP 2 :

**Key Features:**

HTTP 1.1 supports connection reuse i.e. for every TCP connection there could be multiple requests and responses, and pipelining where the client can request several resources from the server at once.

HTTP 2 uses multiplexing, where over a single TCP connection resources to be delivered are interleaved and arrive at the client almost at the same time. It is done using streams which can be prioritized, can have dependencies and individual flow control.

**Authentication Mechanism:**

HTTP 1.1 is relatively secure since it uses digest authentication, NTLM authentication.

In HTTP 2 Security concerns from previous versions will continue to be seen in HTTP/2. However, it is better equipped to deal with them due to new TLS features like connection error of type Inadequate\_Security.

**Protocol Type:**

HTTP 1.1 is Text based protocol that is in the readable form.

HTTP 2 is a binary protocol (HTTP requests are sent in the form of 0s and 1s). Needs to be converted back from binary in order to read it.

**Performance Optimization:**

In HTTP 1.1 Spriting, concatenating, inlining, domain sharding are some of the optimizations used as a workaround to the ‘six connections per host’ rule.

HTTP 2 Removes the need for unnecessary optimization hacks.

1. HTTP VERSION HISTORY:

* Tim Berners -Lee and his team at CERN are credited with inventing the original HTTP, along with HTML and the associated technology for a web server and a text-based web browser. Berners-Lee first proposed the "WorldWideWeb" project in 1989—now known as the World Wide Web.
* The first documented version of HTTP was HTTP V0.9 (1991).
* RFC [1945](https://tools.ietf.org/html/rfc1945) officially introduced and recognized HTTP V1.0 in 1996.
* The HTTP WG planned to publish new standards in December 1995[[16]](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol#cite_note-16) and the support for pre-standard HTTP/1.1 based on the then developing RFC (called HTTP-NG) was rapidly adopted by the major browser developers in early 1996. End-user adoption of the new browsers was rapid
* The HTTP/1.1 standard as defined in RFC 2068 was officially released in January 1997. Improvements and updates to the HTTP/1.1 standard were released under RFC 2616 in June 1999.
* HTTP/2 is a more efficient expression of HTTP's semantics "on the wire", and was published in 2015, and is used by 50.0% of websites.
* HTTP/3 is the proposed successor to HTTP/2, Support for HTTP/3 was added to Cloudflare and Google Chrome in September 2019.

1. BROWSER JS VS NODE JS:

|  |  |
| --- | --- |
| BROWSER JS | NODE JS |
| Javascript is a programming language that is used for writing scripts on the website. | NodeJS is a Javascript runtime environment. |
| Javascript can only be run in the browsers | NodeJS code can be run outside the browser. |
| It is basically used on the client-side. | It is mostly used on the server-side. |
| Javascript is capable enough to add HTML and play with the DOM. | Nodejs does not have capability to add HTML tags. |
| Javascript is used in frontend development. | Nodejs is used in server-side development. |
| Javascript can run in any browser engine as like JS core in safari and Spidermonkey in Firefox. | Nodejs can only run in V8 engine of google chrome. |
| Some of the javascript frameworks are RamdaJS, TypedJS, etc. | Some of the Nodejs modules are Lodash, express etc. These modules are to be imported from npm |

4.What happens when we type the URL in address bar:

* Browser checks cache for DNS entry to find the correspond IP address.
* It looks for following cache. If not found in one, then continues checking to the next until found.

1. Browser Cache
2. Operating Systems Cache
3. Router Cache
4. ISP Cache

* If not found in cache, ISP’s (Internet Service Provider) DNS server initiates a DNS query to find IP address of server that hosts the domain name.  
  The requests are sent using small data packets that contain information content of request and IP address it is destined for.
* Browser initiates TCP connection with the server using synchronize(SYN) and acknowledge(ACK) messages.
* Browser sends an HTTP request to the web server. GET or POST request.
* Server on the host computer handles that request and sends back a response. It assembles a response in some format like JSON,XMLand HTML.
* Server sends out an HTTP response along with the status of response.
* Browser display HTML content.
* Finally, Done.