**TASK 1**

Q. 1) Difference between HTTP1.1 vs HTTP 2

|  |  |
| --- | --- |
| **HTTP 1.1** | **HTTP 2** |
| Keeps all requests and responses in plain text format | Binary framing layer to encapsulate all messages in binary format |
| Won't let us make multiple requests at the same time over the same connection | Faster performance and faster site loading time.  HTTP2 use one connection for parallelism |
| HTTP/1.1 loads resources one after the other, so if one resource cannot be loaded, it blocks all the other resources behind it. | HTTP2 able to use TCP connection and fully multiplexed (download all low images in parallel over the same connection and display as they become available), instead blocking. |
|  | In addition, HTTP2 allows servers to “push” responses proactively into client caches. HTTP/2 uses a more advanced compression method called HPACK that eliminates redundant information in HTTP header packets. |

Q. 2) HTTP version history

HTTP (Hypertext Transfer Protocol) is the underlying protocol of the World Wide Web. HTTP has evolved from the first phase of file exchange, to the modern dimension of the Internet, now carrying high resolution images, videos.

**HTTP/0.9** — (The One-line Protocol)

* Initial version of HTTP — a simple client-server, request-response, tele-net-friendly protocol
* Request nature is single-line (method + path for requested document)
* Connection terminated immediately after the response
* No HTTP headers (cannot transfer other content type files), No status/error codes, No URLs, No versioning

**HTTP/1.0** — (Building extensibility)

* Browser-friendly
* Provided header fields including rich metadata about both request and response (HTTP version number, status code, content type)
* Response not limited to hypertext (Content-Type header provided ability to transmit files other than plain HTML files — e.g. scripts, stylesheets, media)
* Connection terminated immediately after the response

**HTTP/1.1** — (The standardized protocol)

* was published in early 1997
* Pipelining has been added, allowing to send a second request before the answer for the first one is fully transmitted.
* Chunked responses are now also supported.
* Content negotiation, including language, encoding, or type, has been introduced, and allows a client and a server to agree on the most adequate content to exchange.
* The [Host](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Host) header gives the ability to host different domains at the same IP address.

**HTTP/2.0** — (A protocol for greater performance)

* HTTP 2.0 was officially standardized in May 2015.
* It is a binary protocol rather than text.
* It is a multiplexed protocol. Parallel requests can be handled over the same connection.
* It compresses headers. This removes duplication and overhead of data transmitted.
* It allows server push.

Q. 3) List 5 differences between Brower JS vs Node JS

|  |  |
| --- | --- |
| **BrowserJS** | **NodeJS** |
| Used for client side | Is a Server side application |
| Used for writing scripts on websites. | NodeJS is an environment to run the JavaScript. |
| To design / program how the web pages behave on the occurrence of an event. | NodeJS has complete access to the system. This includes read and write directly from the system files. |
| BrowserJS processes response objects. | NodeJS processes request object |
| BrowserJS can run in any engine like Spider monkey (Firefox), JavaScript Core (Safari), V8 (Google Chrome). | Node JS only runs in a V8 engine which mainly used by google chrome and the JS code written under Node JS runs in V8 Engine. |

Q. 4) What happens when you type a URL in address bar in the browser?

1. Enter URL in web browser
2. The browser looks up the IP address for the domain name via DNS
3. Browser checks cache From the Browser, OS, and ISP Cache’s. If not found initiates a query to find IP address of server
4. Once the browser gets the physical address of the server, the browser initiates the TCP connection
5. The browser sends a HTTP request to the server. The HTTP request will be sent using the port 80 as default or port 443 if HTTPS
6. The server sends back a HTTP response
7. The browser begins rendering [HTML parsed DOM tree, CSS parsed CSSOM tree]
8. The layout process begins based upon the blueprint constructed
9. Painting process to display elements according to their structure.