**TASK 1**

**Q.1) Difference between HTTP1.1 vs HTTP2**

i) In HTTP1.1 sometimes make headers much larger increasing the need for some kind of compression whereas in HTTP uses HPACK compression to shrink the size of headers.

ii) In HTTP1.1 stream prioritization is not done whereas in HTTP2 stream prioritization is used which enables developers to gain better control over web page rendering.

iii) In terms of flow control HTTP2 provides a more detailed level of control that opens up the possibility of greater optimization.

iv) HTTP1.1 uses pipelining & head o line blocking method which must make use of multiple TCP connections to lessen the effect of HOL blocking so in HTTP2 its overcomed by use of multiplexing.

v)  HTTP1.1 which keeps all requests and responses in plain text format HTTP2 uses the binary framing layer to encapsulate all messages in binary format, while still maintaining HTTP semantics.

**Q.2) HTTP version history**

**i) HTTP0.9 ( The one line protocol) 1991**

The initial version of HTTP had no version number it has been later called 0.9 to differentiate it from the later versions. HTTP0.9 is extremely simple: requests consist of a single line and start with the only possible method [GET](https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/GET) followed by the path to the resource.

**ii) HTTP1.0 (Building Extensibility) 1996**

The notion of HTTP headers has been introduced both for the requests and the responses allowing metadata to be transmitted and making the protocol extremely flexible and extensible. With the help of the new HTTP headers, the ability to transmit other documents than plain HTML files has been added in these the [Content-Type](https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/Content-Type) header was added.

**iii) HTTP1.1 (The standardized protocol) 1997**

HTTP1.1 was introduced with lots of improvements like A connection can be reused, saving the time to reopen it numerous times to display the resources embedded into the single original document retrieved. Pipelining has been added, allowing to send a second request before the answer for the first one is fully transmitted, lowering the latency of the communication, Additional cache control mechanisms, Content negotiation, including language, encoding, or type, has been introduced. This was all possible by addition of host header; the ability to host different domains at the same IP address now allows server colocation.

**iv) HTTP2.0 ( A protocol for greater performance) 2015**

In HTTP2.0 several addition was done like ALT-Svc, Clients-Hints & Cookies where ALT-Svc allows the dissociation of the identification and the location of a given resource . Client Hints allows the browser or client, to proactively communicate information about its requirements or hardware constraints to the server. Whereas now cookies header now helps to guarantee a secure cookie.

**Q.3) List 5 differences between Browser JS Vs Node JS.**

i) JavaScript is a programming language that is used for writing scripts on the website whereas Node JS is a JavaScript runtime environment.

ii) JavaScript can only be run in the browsers whereas Node JS code can be run outside the browser.

iii) JavaScript is capable enough to add HTML and play with the DOM. whereas Node JS does not have capability to add HTML tags.

iv) JavaScript is used in frontend development. Node JS is used in server-side development.

v) JS is basically used on the client-side & Node JS is mostly used on the server-side.

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

When we type a URL like “[https://example.com”](https://example.com) into your browser the first thing that happens is a Domain Name Server (DNS) matches “[example.com](http://wsvincent.com/)” to an IP address. Then the browser sends an HTTP request to the server and the server sends back an HTTP response. The browser begins rendering the HTML on the page while also requesting any additional resources such as CSS, JavaScript, images, etc. Each subsequent request completes a request/response cycle and is rendered in turn by the browser. Then once the page is loaded some sites (though not mine) will make further asynchronous requests.