TASK 01

1. Difference between HTTP/1.1 and HTTP/2?
   * In HTTP/1.1, flow control relies on the underlying TCP connection. When this connection initiates, both client and server establish their buffer sizes using their system default settings. If the receiver’s buffer is partially filled with data, it will tell the sender its receive window, i.e., the amount of available space that remains in its buffer. This receive window is advertised in a signal known as an ACK packet, which is the data packet that the receiver sends to acknowledge that it received the opening signal. If this advertised receive window size is zero, the sender will send no more data until the client clears its internal buffer and then requests to resume data transmission.
   * HTTP/2 multiplexes streams of data within a single TCP connection. As a result, receive windows on the level of the TCP connection are not sufficient to regulate the delivery of individual streams. HTTP/2 solves this problem by allowing the client and server to implement their own flow controls, rather than relying on the transport layer. The application layer communicates the available buffer space, allowing the client and server to set the receive window on the level of the multiplexed streams.
2. HTTP Version History

|  |  |
| --- | --- |
| HTTTP Version | Year |
| 1.0 | 1996 |
| 1.1 | 1997 |
| 2.0 | 2015 |

1. Difference between browser js and node js
   * Unlike the browser where JavaScript is sandboxed for your safety, node.js has full access to the system like any other native application. This means we can read and write directly to/from the file system, have unrestricted access to the network and can execute software.
   * In the browser, most of the time we interact with the DOM, or other Web Platform APIs like Cookies. Those do not exist in Node.js. We don't have the objects that are provided by the browser.
   * And in the browser, we don't have all the APIs that Node.js provides through its modules, like the filesystem access functionality.
   * in Node.js you control the environment. We know which version of Node.js we will run the application on. Compared to the browser environment, where you don't get the luxury to choose what browser your visitors will use, this is very convenient.
   * Another difference is that Node.js uses the CommonJS module system, while in the browser we are starting to see the ES Modules standard being implemented.
2. What happens when you type a URL in the address bar in the browser?
   * The browser checks the cache for a DNS record to find the corresponding IP address.
   * If the requested URL is not in the cache, ISP’s DNS server initiates a DNS query to find the IP address of the server that hosts it
   * The browser initiates a TCP connection with the server.
   * The browser sends an HTTP request to the webserver.
   * The server handles the request and sends back a response.
   * The server sends out an HTTP response.
   * The browser displays the content.