Guvi Classroom Assignment-1

Name: Shahida Midhat

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

Ans:

HTTP1	HTTP2
Textual Protocol(Status Codes)	Binary Protocol(commands in terms of 1s and 0s are transmitted over wire)
Ordered and blocking(allows a single request/response for every TCP connection.)	Fully multiplexed(multiple requests/responses sent and received asynchronously over a single TCP connection)
Run parallel requests to multiple TCPs for the same Web asset	One connection for parallelism
Headers both for requests (from the client machine) as well as responses (from servers) is used	Allows servers to “push” responses proactively into client caches

Q.2. HTTP version history.

Ans:

1. **HTTP/0.9-[1991]:**This protocol did not use headers and only transmitted plain HTML files. It was a one-line protocol only supporting the GET method.
2. **HTTPS-[1994]:** Netscape Communications created HTTPS (Hypertext Transfer Protocol Secure) to be used with SSL for its web browser, Netscape Navigator.
3. **HTTP/1-[1996]:**The first standardized version of HTTP. Concepts of headers,version information,status codes are used.
4. **HTTP/1.1-[1997]:** Introduced persistent connection,pipelining and cache-control and many other features.
5. **HTTP/2-[2015]:**Based on Google’s introduced SPDY, which supported multiplexing and server push.
6. **HTTP/3-[2019]:** Based on Google’s QUIC, uses UDP instead of TCP.

Q.3. List 5 differences between Browser JS vs Node Js.

Ans.

Browser JS	Node Js
Documents , windows like objects are present	Not includes document ,window, any similar objects.
Uses ES Modules	Uses the CommonJS module system
In this user will decide the environment	In Node.js you can control the environment
Accessing module by using import	Accessing module by using require().
Front-end	Front-end, Backend

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

Ans:

1. You enter a URL in the address bar in the browser
2. The browser looks up for the IP address for the domain name via DNS(Domain Name System)
3. The browser sends a HTTP *request* to the server
4. The server sends back a HTTP *response*
5. The browser begins rendering the HTML(turns content coded in HTML into the text and images for the screen and printer.)
6. The browser sends requests for additional objects embedded in HTML (images, css, JavaScript) and repeats steps 3-5.
7. Once the page is loaded, the browser sends further async requests as needed