DAY 1 :

1. Write a blog on Difference between HTTP1.1 vs HTTP2

**ANSWER:**

HTTP (Hypertext Transfer Protocol) is the backbone of the internet. It's the protocol used by web browsers and servers to communicate and exchange data. With the evolution of the internet and increasing demand for better performance, security, and efficiency, HTTP1.1 was replaced by HTTP2. In this blog, we will explore the key differences between HTTP1.1 and HTTP2 in a table format.

| **Feature** | **HTTP1.1** | **HTTP2** |
| --- | --- | --- |
| Multiplexing | No | Yes |
| Binary Protocol | No | Yes |
| Server Push | No | Yes |
| Header Compression | No | Yes |
| One Connection | Supports one connection at a time | Supports multiple connections |
| Request pipelining | Yes, but with head-of-line blocking | Yes |
| TLS usage | Optional | Mandatory |
| Performance | Slower due to serial processing | Faster due to parallel processing |
| Resource Prioritization | No | Yes |
| Compatibility | Compatible with most browsers | May require updated browsers |

1. Multiplexing: HTTP1.1 allows only one request per connection at a time, which means that requests are processed one after the other. HTTP2, on the other hand, supports multiplexing, which allows multiple requests to be sent over a single connection at the same time. This makes HTTP2 much faster than HTTP1.1, especially when it comes to loading resources such as images, scripts, and stylesheets.
2. Binary Protocol: HTTP1.1 uses plain text to send requests and responses. HTTP2 uses a binary protocol, which makes it more efficient than HTTP1.1 in terms of performance and reduces the overhead of the protocol.
3. Server Push: HTTP2 allows servers to push resources to the client before the client requests them. This means that the server can send resources such as images, stylesheets, and scripts to the client before the client requests them, reducing the page load time.
4. Header Compression: HTTP2 uses HPACK header compression, which reduces the size of the headers sent between the client and the server. This reduces the amount of data that needs to be transferred, improving performance.
5. One Connection vs Multiple Connections: HTTP1.1 allows only one connection per request, which can cause a delay in processing. HTTP2, on the other hand, allows multiple requests to be processed over a single connection, which increases the efficiency of the protocol.
6. Request Pipelining: HTTP1.1 supports request pipelining, which allows multiple requests to be sent over a single connection without waiting for a response. However, it suffers from head-of-line blocking, which means that if one request takes longer to process, it can delay other requests. HTTP2 also supports request pipelining but without the head-of-line blocking problem.
7. TLS Usage: While using Transport Layer Security (TLS) is optional in HTTP1.1, it is mandatory in HTTP2. This means that all HTTP2 traffic is encrypted, which provides better security and privacy.
8. Performance: HTTP2 is faster than HTTP1.1 due to its support for multiplexing, server push, and header compression. This results in faster page load times and better performance.
9. Resource Prioritization: HTTP2 allows for resource prioritization, which means that the server can prioritize the loading of resources based on their importance. This can improve page load times and performance.
10. Compatibility: While HTTP1.1 is compatible with most browsers, HTTP2 may require updated browsers to function properly. However, most modern browsers now support HTTPs
11. **2.** Write a blog about objects and its internal representation in Javascript
12. **2.** Write a blog about objects and its internal representation in Javascript
13. **2 .** Write a blog about objects and its internal representation in Javascript
14. **2 .** Write a blog about objects and its internal representation in Javascript

**2.**

**Answer :**

JavaScript is a versatile and dynamic programming language that is widely used for web development and other applications. One of its key features is its ability to work with objects, which are a fundamental part of the language's syntax and functionality.

In JavaScript, objects are used to represent complex data structures, such as arrays, functions, and classes. An object is essentially a collection of key-value pairs, where each key is a property or method of the object, and the value is the data or functionality associated with that key. Objects can be created using either the object literal notation or the constructor notation, and can be modified or accessed using dot notation or bracket notation.

Internally, objects in JavaScript are implemented using a data structure called a hash table or associative array. A hash table is a type of data structure that allows for constant-time lookups of key-value pairs, making it an efficient and scalable way to store and retrieve data. In JavaScript, the keys in an object's hash table are the properties and methods of the object, and the values are the data or functionality associated with those keys.

In addition to its hash table implementation, objects in JavaScript also have a prototype chain, which is a way of inheriting properties and methods from other objects. When a property or method is accessed on an object, JavaScript first looks for it on the object itself. If it is not found, it then looks for it on the object's prototype, and so on up the prototype chain until it reaches the top-level Object. prototype.

This prototype-based approach to object-oriented programming is a key feature of JavaScript and sets it apart from other languages like Java or C++. It allows for more dynamic and flexible code, as objects can be easily modified or extended at runtime.

Overall, objects are a powerful and essential part of JavaScript, enabling developers to work with complex data structures and implement object-oriented programming paradigms. Understanding how objects are implemented and represented internally in JavaScript can help developers write more efficient and effective code, and take full advantage of the language's rich feature set.

**10 codeketa problems also done**