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

**Ans**

**HTTP/1.1:**

**Connection Handling:** HTTP/1.1 uses a separate connection for each request and response. This means that multiple connections are needed for parallel downloads, which can lead to increased latency.

**Header Compression:** Headers are not compressed in HTTP/1.1, leading to increased overhead. This can be especially problematic for mobile devices or slower network connections.

**Multiplexing:** HTTP/1.1 does not support multiplexing, meaning that only one request can be in progress at a time on a single connection. This limitation can result in a slower user experience, particularly for websites with many resources.

**Prioritization:** There is no built-in mechanism for prioritizing requests in HTTP/1.1. All requests are treated equally, which can lead to suboptimal performance in certain scenarios.

**Server Push:** HTTP/1.1 does not support server push. Server push allows the server to send resources to the client before the client requests them, improving performance by reducing round-trip times.

**HTTP/2:**

**Multiplexing:** HTTP/2 introduces multiplexing, allowing multiple requests and responses to be sent in parallel over a single connection. This significantly improves the efficiency and speed of data exchange.

**Header Compression:** HTTP/2 uses header compression, reducing the amount of overhead associated with sending headers. This is achieved through a mechanism called HPACK, which helps in optimizing the header size.

**Binary Protocol:** HTTP/2 uses a binary protocol as opposed to the text-based protocol of HTTP/1.1. This makes it more efficient to parse and reduces the complexity of communication.

**Prioritization:** HTTP/2 supports stream prioritization, enabling more important resources to be loaded first. This helps improve the overall user experience by ensuring that critical assets are prioritized.

**Server Push:** HTTP/2 introduces server push, allowing the server to send additional resources to the client proactively, without waiting for explicit requests. This can reduce the number of round-trip delays in loading a web page.

1. **Write a blog about objects and its internal representation in Javascript**

**Ans**

**Key-Value Pairs:** JavaScript objects are collections of key-value pairs, where each key is a string or symbol, and the associated value can be of any data type. This makes objects a versatile and flexible data structure

**Properties and Methods:** Properties in JavaScript objects are variables that hold data, while methods are functions associated with an object. Both properties and methods are accessed using the dot notation, like object.property or object.method().

**Prototypes and Inheritance:** JavaScript objects can inherit properties and methods from other objects through a mechanism called prototype-based inheritance. Each object has a prototype object, and if a property or method is not found on the object itself, JavaScript looks up the prototype chain until it finds the property or until the chain ends.

**Object Representation:** Internally, JavaScript engines represent objects using various techniques. One common approach is the use of hash tables to store key-value pairs efficiently. Additionally, objects in JavaScript are dynamic, meaning you can add or remove properties and methods at runtime.

**JSON (JavaScript Object Notation):** Objects in JavaScript can be serialized into a string format using JSON. JSON is a lightweight data interchange format, and it closely resembles the object literal notation in JavaScript. This allows for easy data exchange between a JavaScript application and a server.