Write a blog on Difference between HTTP1.1 vs HTTP2

- Connection Handling: HTTP/1.1 uses a single connection per request/response cycle, while HTTP/2 uses a single connection per client-server session, allowing for multiple requests and responses to be sent over the same connection.
- Header Compression: HTTP/1.1 requires headers to be sent with every request and response, while HTTP/2 compresses headers, reducing their size significantly, leading to smaller requests and responses.
- Server Push: HTTP/2 introduced server push, which allows the server to send resources to the client before they are requested, leading to faster page load times. HTTP/1.1 does not support server push.
- Multiplexing: HTTP/2 allows for multiple requests and responses to be sent over the same connection simultaneously, reducing the number of connections required and improving performance. HTTP/1.1 only allows for one request/response cycle at a time.
- Security: HTTP/2 requires the use of SSL/TLS encryption for all connections, adding an extra layer of security. HTTP/1.1 does not require SSL/TLS encryption, making it susceptible to interception and eavesdropping.

Write a blog about objects and its internal representation in Javascript

- Objects in JavaScript are implemented as dynamic associative arrays, also known as hash maps or dictionaries. This means that they can be used to store any type of value, and the properties can be added or removed at runtime.
- In JavaScript, every object has an internal [[Prototype]] property, which points to another object. This is known as the prototype chain, and it allows objects to inherit properties and methods from other objects. The prototype chain is created when objects are created using the new keyword or by using object literals.
- JavaScript objects can have properties with different attributes, such as writable, enumerable, and configurable. These attributes control how the property can be accessed, modified, and deleted. For example, setting a property to be non-writable prevents its value from being changed.
- When an object is created in JavaScript, it is allocated in memory as a collection of properties and their values. The properties are stored in a hash table, and the values are stored in the object's internal slots. The size of an object in memory depends on the number and type of its properties.
- JavaScript objects have several internal methods, such as [[Get]], [[Set]], and [[Call]]. These methods are used to get and set property values, call object

methods, and perform other operations on objects. These methods are invoked when an object is accessed or manipulated in code