Task

Q1) HTTP/1.1 & HTTP/2

HTTP/1.1:

The HTTP/1.1 is developed by Tim Berners-Lee (Timothy Berners-Lee) in 1989 as a communication standard for the World Wide Web. The first usable version of HTTP/1.1 is created in 1997. The delay was because it went through several stages of development. The HTTP/1.1 is still used on the web. Tim Berners-Lee is the founder of WWW (World Wide Web). HTTP stands for Hypertext transfer protocol. HTTP is basically an application protocol which acts as the bridge between the client computer and remote web server. That is HTTP helps to transfer the information between the client computer and web server. Here the communication happens like the user or the client computer asks or requests something through either the GET method or POST method. So the web server replies with something like an HTML page back to the client computer where all the content requested by the user will be there. The major problem in HTTP/1.1 is it loads the resources one after another. So, if one resource cannot be loaded, it blocks all the other resources behind it. Header compression is used by HTTP/1.1 to speed up the web performance as small files loads faster than the larger files.

HTTP/2:

In 2015, a new version of HTTP was created and it is called HTTP/2. The creator of HTTP/1.1 was not ambitious about the protocol. Their primary aim was to create a protocol which helps to create a communication between client and web server. So HTTP/2 is way faster and more efficient than HTTP/1.1. One way the HTTP/2 is faster is in how it prioritizes the content during the loading process. Prioritization basically means how much priority or in what order the priority should be given to the contents in the web page requested by the user. If the user loads the page, the elements in the header will be getting more priority than the elements in the footer. This helps to render the web page smoothly. HTTP/2 allows developers to decide which page resources should load first every time, and this feature is called weighted prioritization. HTTP/2 is able to use a single TCP connection to send multiple streams of data at once so that no resource block any other resources behind it. It is done by splitting the data into binary code messages and numbering it so that the client computer or browser knows how to arrange the binary messages properly. Here HTTP/2 supports server push, which basically means the server pushes the content to the client before the client asks for it. The server also sends a message where what are the contents pushed by the server and what contents available in the local machine which the user or client can expect. HTTP/2 uses an advanced compression method called HPACK that eliminates unwanted information from HTTP header packets. So this will helps to load the web pages faster than HTTP/1.1 even though it had header compression.

HTTP/1.1	HTTP/2
HTTP/1.1 HTTP/2 Slower compared to HTTP/2.	Faster.
Relies on TCP/IP connection.	Relies on TCP/IP connection.
Loads resources one after another.	Loads resources parallelly.
Server push is not available.	Server push is available.
Basic Header compression.	Advanced Header compression.
The Default Communication method is GET.	No default communication method.

Q.2) Objects and its internal representation in JavaScript.

Objects are the representation of real-world entities in any language representing things by defining its properties along with their values. In JavaScript, objects may be defined as an unordered collection of related data, of primitive or reference types, in the form of "key: value" pairs.

Ways to create an object in JavaScript are as follows-

Object literal

Object literal is a comma-separated list of name-value pairs wrapped in curly braces. Object literals encapsulate data, enclosing it in a tidy package.

var car= {id:1 , name:'abc' , display: function() }

As evident from the above example property values can be of any data type, including array literals, functions, nested object literals, or primitive data type.

Object.create()

The method creates a new object, using an existing object as the prototype of the newly created object.

using the object literal example as prototype-

var car2 = Object.create(car);

car.id=2;

car.name='xyz';

Object constructor

Useful when we require to create multiple objects of similar type. In this case, a constructor (kind of blueprint) is created and multiple objects can be initialized using the new keyword using the constructor as a wrapper for the newly created objects.

construction function-

```
function Person(name, age, eye) {
this.Name = name;
this.age = age;
this.eyeColor = eye;
}
```

Creating objects using constructor-

```
var p1= new Person("John", 50, "blue");
var p2= new Person("Sally", 48, "green");
```

Object.assign()

It is used to copy the values and properties from one or more source objects to a target object. It invokes getters and setters since it uses both Get on the source and Set on the target.

Eg.

Input :

```
var obj1 = { a: 10 };
var obj2 = { b: 20 };
var obj3 = { c: 30 };
var new_obj = Object.assign(o1, o2, o3);
console.log(new_obj);
Output : Object { a: 10, b: 20, c: 30 }
```

Object.fromEntries

This method transforms a list of key-value pairs into an object.

```
const entries = new car([
['id', 4],
['color', 'blue']
]);

const car1= Object.fromEntries(entries);

console.log(car1);
output: Object { id: 4, color: 'blue'}
```

Unlike other object-oriented programming languages, javascript doesn't have classes instead of that javascript is a prototype-based language allowing all the functionalities as in other class-based programming languages like JavaScript allows you to create hierarchies of objects and to have the inheritance of properties and their values and all this is done mainly using the constructor functions.