Javascript

1. **What is javascript? How does it work?**
2. **What is an asynchronous operation ? Is Javascript synchronous or asynchronous.**
3. **Explain async using set timeout.**
4. **JS a single threaded or multi threaded ?**
5. **Is JS an asynchronous or synchronous language ?**
6. **Difference between var let and const**
7. **What is hoisting**
8. **What is a promise? What are the states of promises? What is promisify?**
9. **What is async await?**
10. **Difference between promise and async & await. Is async await notation preferred over promises? If so, why?**
11. **What are JavaScript Objects ? how to create them.**
12. **How to search for a value or a key in an object?**
13. **What is a callback function?**
14. **What are the drawbacks of callback and what is callback hell.**
15. **What is the difference between callback and promise. How can you write the same async operation using promise and callback**
16. **Difference between function and arrow function**
17. **Difference between i++ and ++i**
18. **What is the “this” operator?**
19. **What is the closure concept?**
20. **what is closure - with code, why is it required, it's benefits**
21. **What is json? What do the functions json.parse and json.stringify do?**
22. **What is the difference between '==' and '===' operators?**
23. **What is the type of operator? What is NaN?**
24. **What are set timeout and set interval functions and what is the difference between them?**
25. **What is functional scope and block scope?**
26. **What is shallow Copy and deep Copy**
27. **What is the difference between map and filter. Explain math, the built-in javascript object, functionalities.**
28. **What is ES6? Explain some features of ES6**
29. **Difference btw typescript & JavaScript.**
30. **What is regex?**
31. **what is first class functions**
32. **diff between primitive and reference data types**
33. **Difference between null and undefined**
34. **difference between slice and splice**
35. **what is spread operator and rest operator**
36. **how to convert integer to string and string to integer**
37. **Explain JS Object methods like- Object. Keys, Object. Values and Objects. Entries**

API

1. **What is an API?**
2. **What is Rest Api?**
    a. **What are the rest of the Api ? Why are they stateless?**
3. **What are the different ways to pass data to an API while requesting a Create operation? Discuss the Content-Type header in brief**
4. **Explain the GET, POST, PUT, POST and PATCH method types**
5. **DIfference between PUt and Patch**
6. **What is an HTTP status code? Discuss the popularly used codes along with their use case.**
    a. **What are http status codes? Please elaborate?**

Nodejs

1. **What is Nodejs? What is the difference between Javascript and Nodejs?**
2. **Node js is single threaded or multi threaded? Explain in detail what that means.**

3. **What is meant by request and response cycle in a nodejs application?**
4. **What is express? What are the benefits of using this framework?**
5. **What is Middleware? What are global and route level middlewares?**
    a. **Explain about application and route based middleware?**
6. **What is the function of an express router? What is a route and route splitting?**
7. **How is Node Js different from a multithreaded architecture?**
8. **What is .env file or environment variables? Why are they important?**
9. **What is an event loop? Explain with an example.**
10. **security - <u>how would you protect your node js server?</u>**

Mongodb

1. **Why is a database? Name a few examples.**
2. **Why do you use MongoDB over MySQL and PostgreSQL? MySql vs MongoDb (or SQL/RDBMS vs NoSql).**
3. **Compare relational and non-Relational databases and their benefits or drawbacks?**
4. **What kind of db is MongoDb? Why do we define schema in nodejs application when working with Mongodb if it is schema-less?**
5. **What is an aggregation pipeline? What are some stages used in it?**
6. **What is a Schema and a model in mongoose? How are they different?**
7. **Discuss some common mongoose APIs for CRUD operations.**
8. **Why do you have to use await if you want to process the result?**
9. **What are indexes? What is the default index value set by MongoDB?**
    a. **<u>What do you mean by database indexing?</u>**
10. **How to link the document of one collection to a document in another collection using mongoose. Explain reference in mongoose.**
11. **What is the difference between find, findOne, findOneAndUpdate and findById APIs.**
12. **Allso between updateOne, update and updateMany? When should you use each?**

AWS

1. **What is AWS and AWS S3 ?**
2. **What are the commands to push in AWS S3?**
3. **What is meant by a bucket?**
4. **Can we upload images in the database w/o using AWS S3. What are the benefits of using s3 instead?**

Caching

1. **What is meant by cache and caching?**
2. **Why we use redis**
3. **Why do we need redis if we have databases?**
    a. **Redis advantages and disadvantages.**
4. **Explain cache miss and cache hit.**

Auth

1. **What is authentication and authorization? What is the difference between them?**
2. **What is jwt? Describe the parts of JWT? What are the benefits of using a JWT**
    a. **Why is authorization necessary?**
3. **How did you protect your Routes?**

Project

1. **How do you implement authentication in your project?**
2. **Explain the flow how a user is authenticated and the prerequisite for the same (token generation and sharing with the client)**

    a. **How to generate a token ? How to verify it ?**
3. **Explain URL shortener project (explain the overview, features and how using redis). What is shortId. What happens if the same long url is passed in the request twice. How can you create a unique short code without using the shortId package? How is the redirection happening?**
4. **Why do we use mvc (model view controller) in projects**
5. **Deploy your project on Heroku ?**

Other (OOPS + Git + Multithreading)

1. **OOPS overview.**
2. **What is inheritance and abstraction?**
3. **What is Git?**
4. **What are some basic git commands?**
5. **What is a Git repository and branches?**
6. **What is difference between Git and GitHub**
7. **What is class,id in html? What is DOM?**

JavaScript

1.What is javascript? How does it work?
Ans 1.

JavaScript is a text-based programming language used both on the client-side and server-side that allows you to make web pages interactive. Where HTML and CSS are languages that give structure and style to web pages, JavaScript gives web pages interactive elements that engage a user.

JavaScript is a single-threaded programming language, which means it has a single Call Stack. Therefore it can do one thing at a time.

The Call Stack is a data structure which records basically where in the program we are. If we step into a function, we put it on the top of the stack. If we return from a function, we pop off the top of the stack. That's all the stack can do.

2. What is an asynchronous operation? Is JavaScript synchronous or asynchronous?
Ans 2.

Synchronous: As the name suggests synchronous means to be in a sequence, i.e., every statement of the code gets executed one by one. So, basically a statement has to wait for the earlier statement to get executed.

Asynchronous: Asynchronous code allows the program to be executed immediately where the synchronous code will block further execution of the remaining code until it finishes the current one. This may not look like a big problem but when you see it in a bigger picture you realize that it may lead to delaying the User Interface.

For example, if you are loading an image from a server, the JavaScript code will not wait for the image to load before continuing. Instead, the code will continue to execute while the image is loading in the background.

Asynchronous programming is a technique that enables your program to start a potentially long-running task and still be able to be responsive to other events while that task runs, rather than having to wait until that task has finished. Once that task has finished, your program is presented with the result.

JavaScript is Synchronous, blocking, single-threaded language. That just means that only one operation can be in progress at a time.

3. Explain async using set timeout.
Ans 3.

setTimeout() is an asynchronous function, meaning that the timer function will not pause execution of other functions in the functions stack. In other words, you cannot use setTimeout() to create a "pause" before the next function in the function stack fires.

or /- The `setTimeout()` function can be used to execute a function after a specified amount of time. This can be used to create asynchronous operations
For example, the following code will execute the function `myFunction()` after 5 seconds:

```
setTimeout(myFunction, 5000);
```

4. Is JS a single threaded or multi threaded ?
Ans4.

JavaScript is a single-threaded language. This means that JavaScript code can only execute one task at a time. However, JavaScript is asynchronous, which means that it can handle multiple tasks by running them in parallel.

Javascript is a single threaded language. This means it has one call stack and one memory heap. As expected, it executes code in order and must finish executing a piece code before moving onto the next.

5. Is JS an asynchronous or synchronous language ?
Ans5.

Well, Javascript is Synchronous, It is a synchronous, single-threaded language. Because it has a single thread that's why it can only execute one command at a time and the other commands need to wait for executing before the running command executes. And the term synchronous means one at a time.

6. Difference between var let and const.
Ans6.

	var	let	const
Hoisting	Hosted at top of global scope. Can be used before the declaration.	Hosted at top of some private scope and only available after assigning value. Can not be used before the declaration.	Hosted at top of some private scope and only available after assigning value. Can not be used before the declaration.
Scope	Global scope normally. Start to end of the function inside of the function.	Block scoped always. Start to end of the current scope anywhere.	Block scoped always. Start to end of the current scope anywhere.
Redeclaration	Yes, we can redeclare it in the same scope.	No, we can't redeclare it in the same scope.	No, we can't redeclare or reinitialize it.

7. What is hoisting ?
Ans7.
Hoisting is a JavaScript feature that moves variables and function declarations to the top of their scope before they are actually used. This means that you can access a variable or function before it is declared, as long as it is declared somewhere in the scope.

or

Hoisting in JavaScript is a behavior in which a function or a variable can be used before declaration. Variable Hoisting - In terms of variables and constants, keyword "var" is hoisted and "let" and "const" does not allow hoisting.

8. What is a promise? What are the states of promises? What is promisify?
Ans8.

In JavaScript, callback functions were initially used to handle asynchronous operations. However, callbacks were limited in terms of functionality and often led to confusing code, so, promises were introduced to cater to these problems. According to MDN, "the Promise object represents the eventual completion (or failure) of an asynchronous operation and its resulting value."

A promise object has one of three states:

- **pending: is the initial state.**

- fulfilled: indicates that the promised operation was successful.
- rejected: indicates that the promised operation was unsuccessful.

Promisify() method defined in the utilities module of Node. js standard library. It is basically used to convert a method that returns responses using a callback function to return responses in a promise object

or

A promise is a JavaScript object that represents the eventual completion of an asynchronous operation. Promises have a number of states, including:

- `pending`: The promise has not yet been resolved or rejected.
- `resolved`: The promise has been resolved with a value.
- `rejected`: The promise has been rejected with an error.

The `promisify` function can be used to convert a function that returns a value to a promise. This allows you to use promises with functions that were not originally designed to be asynchronous.

9. What is async await?

Ans9.

We use the async keyword with a function to represent that the function is an asynchronous function. The async function returns a promise.
The await keyword is used inside the async function to wait for the asynchronous operation.

10. Difference between promise and async & await. Is async await notation preferred over promises? If so, why?

Ans10.

Promise	Async/Await
• Promise is an object representing an intermediate state of operation which is guaranteed to complete its execution at some point in future.	• Async/Await is a syntactic sugar for promises, a wrapper making the code execute more synchronously.
• Promise has 3 states – resolved, rejected and pending.	• It does not have any states. It returns a promise either resolved or rejected.
• Error handling is done using .then() and .catch() methods.	• Error handling is done using .try() and .catch() methods.
• Promise chains can become difficult to understand sometimes.	• Using Async/Await makes it easier to read and understand the flow of the program as compared to promise chains.
• If the function “fxn1” is to be executed after the promise, then promise.then(fxn1) continues execution of the current function after adding the fxn1 call to the callback chain.	• If the function “fxn1” is to be executed after await, then await X() suspends execution of the current function and then fxn1 is executed.

Yes, async await notation is preferred over promises because, using Async/Await makes it easier to read and understand the flow of the program as compared to promise chains.

11. What are JavaScript Objects ? how to create them.

Ans11.

A javaScript object is an entity having state and behavior (properties and method). For example: car, pen, bike, chair, glass, keyboard, monitor etc. JavaScript is an object-based language. Everything is an object in JavaScript.

There are different ways to create new objects:

- Create a single object, using an object literal.
- Create a single object, with the keyword new.
- Define an object constructor, and then create objects of the constructed type.

- **Create an object using Object.create().**

12. How to search for a value or a key in an object?
Ans12.

Object.keys(obj) – returns all the keys of object as array
Object.values(obj) – returns all the values of the object as array
Object.entries(obj) – returns an array of [key, value]

13. What is a callback function?
Ans13.

Any function that is passed as an argument to another function so that it can be executed in that other function is called as a callback function.

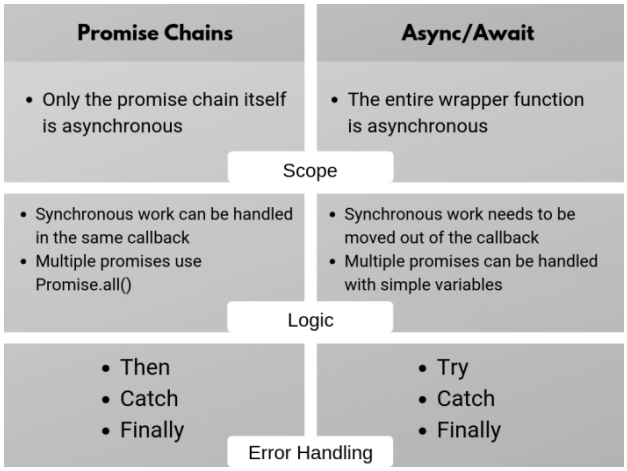
14. What are the drawbacks of callback and what is callback hell.
Ans14.

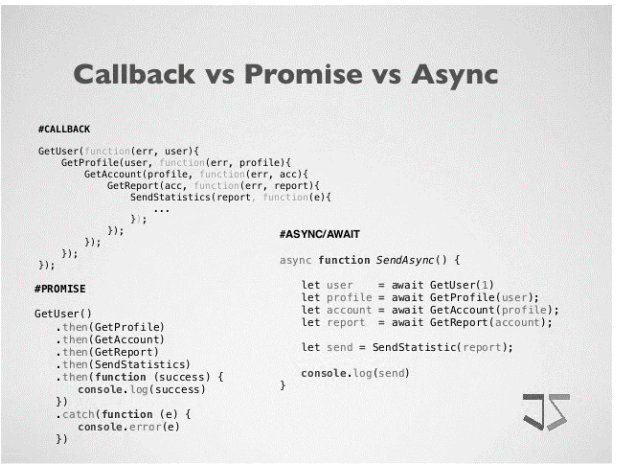
The main disadvantage of using callback is Callback Hell. Callback hell is essentially nested callbacks where each callback is dependent on previous callback. Callback hell affects the readability and maintainability of the code.

Callback Hell **: Callback Hell is essentially nested callbacks stacked below one another forming a pyramid structure. Every callback depends/waits for the previous callback, thereby making a pyramid structure that affects the readability and maintainability of the code**

15. What is the difference between callback and promise and Async-Await? How can you write the same async operation using promise and callback .
Ans14.

- **Callbacks are functions passed as arguments into other functions to make sure mandatory variables are available within the callback-function's scope.**
- **Promises are placeholder objects for data that's available in the future. As soon as their state changes from pending to resolved, .then() method can be called to make the data available for subsequent operations.**
- **Async / Await is syntactic sugar over promises. Instead of using .then(), you can assign data that's resolved by a promise to a variable available within an async function() scope.**





16. Difference between function and arrow function.
Ans16.

Arrow function is one of the features introduced in the ES6 version of JavaScript. It allows you to create functions in a cleaner way compared to regular functions.

Since regular functions are constructible, they can be called using the new keyword. However, the arrow functions are only callable and not constructible, i.e arrow functions can never be used as constructor functions. Hence, they can never be invoked with the new keyword.

17 .Difference between i++ and ++i
Ans17.

Postfix increment (i++) increments, but returns the old value (i.e., the value before the increment).
Prefix increment (++i) increments, and returns the new, incremented value.

18. What is the "this" operator?
Ans18.

In JavaScript, this keyword refers to an object. Which object depends on how this is being invoked (used or called).

19. What is the closure concept?
Ans19.

A closure is the combination of a function bundled together (enclosed) with references to its surrounding state (the lexical environment). In other words, a closure gives you access to an outer function's scope from an inner function. In JavaScript, closures are created every time a function is created, at function creation time.

20. What is closure - with code, why is it required, its benefits.
Ans20.

A closure is when a function remembers and has access to variables in its lexical / outer scope even when the function is executed outside of its lexical scope. Closures are created at function creation time. Variables are organized into units of scope, such as block scope or function scope.
Some advantages of closures:

- **Variables in closures can help you maintain a state that you can use later.**
- **They provide data encapsulation.**
- **They help remove redundant code.**
- **They help maintain modular code.**

21. What is json? What do the functions json.parse and json.stringify do?
Ans21.

JavaScript Object Notation (JSON) is commonly used for transmitting data in web applications
parse() is used for parsing data that was received as JSON; it deserializes a JSON string into a JavaScript object. JSON. stringify() on the other hand is used to create a JSON string out of an object or array; it serializes a JavaScript object into a JSON string.

imp question - Why Using Promises or Async-await is Better Than Callbacks? and Why Using Async-await is Better Than Promises?

► Using callbacks is good, but as developers we need to know how to avoid callback hell (nested functions). Callback hell is a very bad case where you have a lot of callbacks inside of callbacks. Callback hell can make code not readable nor maintainable. Below are some ways to avoid callback hell:

- **Modularization: break callbacks into independent functions.**

- **You can write callbacks using promises. Promises break down the code and chain them together. So instead of code indenting to the right like in callbacks, your codes are now increased in vertical direction. Which solved the major problem of callback hell.**
- **You can write callbacks using the new async/await. Async await removes the .then blocks of your code, replacing them with try and catch blocks to make your code look synchronous.**

► Async/await is equal to non-blocking, just like promises. However, the purpose of using async-await functions is to simplify the behavior of promises synchronously, and to perform behaviors on a group of promises. Below are some advantages of using Async/await over promises:

- **Lower memory requirements.**
- **Clean and neat syntax.**
- **More readable.**
- **Handles conditionals smoother and cleaner than promises.**
- **Handling and reporting errors better.**

22. What is the difference between '==' and '===' operators?

Ans22.

“==” – It is used to compare two values (“equal to”)

“===” – It is also used to compare two value but also compare its datatype (“equal value and equal type”)

E.g.

```
let a = 10, b = "10"
a==b ? console.log(true) : console.log(false)        // true
a===b ? console.log(true) : console.log(false)       // false
```

23. What is the type of operator? What is NaN?

Ans23.

Typeof in JavaScript is an operator used for type checking and returns the data type of the operand passed to it. The operand can be any variable, function, or object whose type you want to find out using the typeof operator.

NaN is a short form of ‘Not a Number”. NaN values are generated when arithmetic operations result in undefined or unrepresentable values.

24. What are set timeout and set interval functions and what is the difference between them?

Ans 24.

- **setTimeout allows us to run a function once after the interval of time.**
- **setInterval allows us to run a function repeatedly, starting after the interval of time, then repeating continuously at that interval.**

`setTimeout()`:

It is a function that execute a JavaScript statement `AFTER` x interval.

```
setTimeout(function () {
    something();
}, 1000); // Execute something() 1 second later.
```

`setInterval()`:

It is a function that execute a JavaScript statement `EVERY` x interval.

```
setInterval(function () {
    somethingElse();
}, 2000); // Execute somethingElse() every 2 seconds.
```

The interval unit is in `millisecond` for both functions.

25. What is functional scope and block scope?

Ans 25.

Function Scope: When a variable is declared inside a function, it is only accessible within that function and cannot be used outside that function.

Block Scope: A variable when declared inside the if or switch conditions or inside for or while loops, are accessible within that particular condition or loop. To be concise the variables declared inside the curly braces are called as within block scope.

26. What is shallow Copy and deep Copy?

Ans 26.

Shallow copy	Deep copy
Shallow copy is faster	Deep copy is slower as compared to shallow copy
It stores the references of an object to the original memory address	It only stores the copy of object's value.
It reflects changes made in copied object to original object	It does not reflect any change to the new object
Both copied and original object point to same memory location	Copied and original object does not point to same memory location.

27. What is the difference between map and filter. Explain math, the built-in JavaScript object, functionalities.

Ans 27.

Map

The map() method is used for creating a new array from an existing one, applying a function to each one of the elements of the first array.

Syntax

```
var new_array = arr.map(function callback(element, index, array) {
    // Return value for new_array }
```

In the callback, only the array element is required. Usually, some action is performed on the value and then a new value is returned.

Filter

The filter() method takes each element in an array and it applies a conditional statement against it. If this conditional returns true, the element gets pushed to the output array. If the condition returns false, the element does not get pushed to the output array.

Syntax

```
var new_array = arr.filter(function callback(element, index, array) {
   // Return true or false }
```

The syntax for filter is similar to map, except the callback function should return true to keep the element, or false otherwise. In the callback, only the element is required.

Math

Math is a built-in object that has properties and methods for mathematical constants and functions. It's not a function object.

Math works with the Number type. It doesn't work with BigInt.
Unlike many other global objects, Math is not a constructor. All properties and methods of Math are static.

The *syntax* for Math any methods is: Math.method(number)

28. What is ES6? Explain some features of ES6.

Ans 28.

ES6 or the ECMAScript 2015 is the 6th and major edition of the ECMAScript language specification standard. It defines the standard for the implementation of JavaScript and it has become much more popular than the previous edition ES5.

ES6 comes with significant changes to the JavaScript language. It brought several new features like, let and const keyword, rest and spread operators, template literals, classes, modules and many other enhancements to make JavaScript programming easier and more fun.

Most popular ES6 features that we can use in your everyday JavaScript coding.

- **let and const Keywords**
- **Arrow Functions**
- **Multi-line Strings**
- **Default Parameters**
- **Template Literals**
- **Destructuring Assignment**
- **Enhanced Object Literals**
- **Promises**
- **Classes**
- **Modules**

29. Difference btw typescript & JavaScript.

Ans 29.

TypeScript is an object-oriented programming language developed by Microsoft Corporation, whereas JavaScript is the programming language for the web.

TypeScript is an open-source language to build large-scale web apps, whereas JavaScript is a server-side programming language that helps to develop interactive web pages.

When JavaScript was developed, the JavaScript development team introduced JavaScript as a client-side programming language. But as people were using JavaScript, developers also realized that JavaScript could

be used as a server-side programming language. However, as JavaScript was growing, JavaScript code became complex and heavy. Because of this, JavaScript wasn't even able to fulfill the requirement of an Object-Oriented Programming language. This prevented JavaScript from succeeding at the enterprise level as a server-side technology. So, TypeScript was created by the development team to bridge this gap.

- TypeScript is known as an Object-oriented programming language whereas JavaScript is a prototype-based language.
- TypeScript has a feature known as Static typing but JavaScript does not support this feature.
- TypeScript supports Interfaces but JavaScript does not.

30. What is regex?

Ans 30.

Regex is well known as a regular expression, is a pattern of characters. The pattern is used to do pattern-matching "search-and-replace" functions on text.

In JavaScript, a RegExp Object is a pattern with Properties and Methods.

31. what is first class functions

Ans 31.

A programming language is said to have First-class functions if functions in that language are treated like other variables. So, the functions can be assigned to any other variable or passed as an argument or can be returned by another function. JavaScript treats functions as first-class-citizens. This means that functions are simply a value and are just another type of object.

32. Diff between primitive and reference data types

Ans 32.

Whenever you create a variable in JavaScript, that variable can store one of two types of data, a primitive value or a reference value. If the value is a number, string, boolean, undefined, null, or symbol, it's a primitive value. If it's anything else (i.e. typeof object), it's a reference value.

Primitive Values	Reference Values
number	anything that is
string	"typeof" "object"
Boolean	objects
undefined	arrays
null	functions
symbol	

33. Difference between null and undefined

Ans 33.

- In JavaScript, undefined means a variable has been declared but has not yet been assigned a value.
- null is an assignment value. It can be assigned to a variable as a representation of no value

undefined is a type itself (undefined) while null is an object.

```
null === undefined    // false
null == undefined     // true
null === null         // true
```

34. Difference between slice and splice.

Ans 34.

slice

The term 'slice' literally means to cut something into pieces. It is used to cut out elements from an array. It does not affect the original array.

splice

'Splice', according to the dictionary, means to join things together. It is used to add or remove elements from an array or replace them.
slice returns a piece of the array but it doesn't affect the original array. splice changes the original array by removing, replacing, or adding values and returns the affected values.

When you use each one is up to you. If you have to alter the content of the array, splice is the way to go, but if you just want to obtain a subarray, slice should be your choice.

35 . What is the spread operator and rest operator?

Ans 35.

Spread Operator

The JavaScript spread operator (...) allows us to quickly copy all or part of an existing array or object into another array or object.

Ex- const numbersOne = [1, 2, 3];
const numbersTwo = [4, 5, 6];
const numbersCombined = [...numbersOne, ...numbersTwo];

Output- 1,2,3,4,5,6

Rest Operator

The rest operator is used to put the rest of some specific user-supplied values into a JavaScript array.
So, for instance, here is the rest syntax: ***...yourValues***

The three dots (...) in the snippet above symbolize the rest operator.
The text after the rest operator references the values you wish to encase inside an array. You can only use it before the last parameter in a function definition.

36. How to convert integer to string and string to integer

Ans 36.

In JavaScript ,

- **The toString() function is used to convert the integer (number) to string.**
- **The parseInt() function (or a method) is used to convert the string to integer (number), parseInt() function returns Nan(not a number) when the string doesn't contain a number.**

37. Explain JS Object methods like- Object. Keys, Object. Values and Objects. Entries

Ans 37.

Object.keys() - Object.keys() creates an array containing the keys of an object.
Object.values() - Object.values() creates an array containing the values of an object.
Object.entries() - Object.entries() creates a nested array of the key/value pairs of an object.

Nodejs

1. Why is nodejs? What is the difference between javascript and nodejs?
Ans 1.

Node.js is a bridge, open-source Js runtime environment. JavaScript code can now execute outside of the browser. Node.js has many components and is primarily used for web development.

JavaScript	Node.js
It is an accessible, bridge, parsed, lightweight, reactive, and web apps programming language.	It's a bridge, open-source Js runtime environment for executing Js on the server.
It's a programming language, after all. Any browser with a competent browser engine will operate.	It's a JavaScript translator and environment that includes some valuable libraries for JavaScript programming.
It's most commonly used on client-side servers.	It's mainly popular on the server-side.
The node community does not care about JavaScript.	All node projects represent the JavaScript community.
It's made for creating network-centric apps.	It's made for real-time data-intensive apps that run on multiple platforms.
It's a new release of the ECMA script that works on the C++-based V8 engine.	C++, C, and JavaScript are used.
TypedJS, RamdaJS, and other JavaScript frameworks are examples.	Nodejs modules include Lodash and Express. All of these modules must be imported from npm.

2. Node js is single threaded or multi threaded? Explain in detail what that means.
Ans 2.

Node.js is single-threaded but the catch is that there are some libraries in Node.js that are not single-threaded.

Event loop runs one process at a time. That means it can only execute one function at a time, and since functions can have multiple instructions, the event loop will execute one instruction at a time.

3. What is meant by request and response cycle in a nodejs application?
Ans 3.

The request module is used to make HTTP calls. It is the simplest way of making HTTP calls in node. js using this request module.
The response object represents the HTTP response that an Express app sends when it gets an HTTP request.
4. What is express? What are the benefits of using this framework?
Ans 4.

ExpressJS is a prebuilt NodeJS framework that can help you in creating server-side web applications faster and smarter.

Advantages of express

1) **Makes Node.js web application development fast and easy.**
2) **Easy to configure and customize.**
3) **It allows you to define routes of your application based on HTTP methods and URLs.**
4) **Includes various middleware modules that you can use to perform additional tasks on request and response.**
5) **Easy to integrate with different template engines like Jade, Vash, EJS etc.**
6) **It allows you to define an error handling middleware.**
7) **Easy to serve static files and resources of your application.**
8) **It allows you to create a REST API server.**
9) **Easy to connect with databases such as MongoDB, Redis, MySQL**

5. What is Middleware? What are global and route level middlewares?

Ans 5.

Middleware functions are functions that have access to the request object (req), the response object (res), and the next middleware function in the application's request-response cycle.
The next middleware function is commonly denoted by a variable named next.

10.Explain about application and route based middleware?

Ans.

The function is executed every time the app receives a request.(application-level)
A piece of code that comes in the middle of request and response .(route-level)

11.What is the function of an express router? What is a route and route splitting?

Ans.

Router() function is used to create a new router object. This function is used when you want to create a new router object in your program to handle requests.
Route-based code-splitting is a method of splitting React components that involves using dynamic import() on lazy load route components.

12.How Node Js is different from a multithreaded architecture?

Ans.

Node JS Platform does not follow the Request/Response Multi-Threaded Stateless Model. It follows Single Threaded with Event Loop Model.

13.What are the .env file or environment variables? Why are they important?

Ans.

Environment variables in Node are used to store sensitive data such as passwords, API credentials, and other information that should not be written directly in code.

14.What is an event loop? Explain with an example.

Ans.

JavaScript has a runtime model based on an event loop, which is responsible for executing the code, collecting and processing events.
Event loop is an endless loop, which waits for tasks, executes them and then sleeps until it receives more tasks.

15. Security - how would you protect your node js server?

Ans.

- Source code security.
- Authentication, authorization, and encryption/hashing algorithms.
- Helmet middleware.
- CORS.
- Protection against CSRF, SQLi, and DoS/DDoS.
- Storage of sensitive credentials.
- Deployment and hosting environment.
- SSH, firewalls, honeypots, VPCs.

AWS

1. What is AWS and AWS S3 ?
Ans 1.

AWS (Amazon Web Services) is a comprehensive, evolving cloud computing platform provided by Amazon that includes a mixture of infrastructure as a service, platform as a service and packaged software as a service offerings.

Amazon S3 or Amazon Simple Storage Service is a service offered by Amazon Web Services (AWS) that provides object storage through a web service interface.Amazon S3 can store any type of object, which allows uses like storage for Internet applications, backups, disaster recovery, data archives, data lakes for analytics, and hybrid cloud storage

2. What are the commands to push in AWS S3?
Ans 2.

In the Amazon S3 console, choose the bucket where you want to upload an object, choose Upload, and then choose Add Files. In the file selection dialog box, find the file that you want to upload, choose it, choose Open, and then choose Start Upload.

3. What is meant by a bucket?
Ans 3.

A bucket is a container for objects. To store your data in Amazon S3, you first create a bucket and specify a bucket name and AWS Region.

4 .Can we upload images in the database w/o using AWS S3. What are the benefits of using s3 instead?
Ans 4.

No, we cannot upload image in database without using AWS S3.

Benefits of using S3
1. Budget-friendly
2. High scalability
3. Durability
4. High availability
5. Security
6. Easy to manage

Caching

1. What is meant by cache and caching?
Ans 1.

A cache is a high-speed data storage layer which stores a subset of data, typically transient in nature, so that future requests for that data are served up faster than is possible by accessing the data's primary storage location. Caching allows you to efficiently reuse previously retrieved or computed data.

2. Why do we use redis .

Ans 2.

Redis can be used to store your website sessions to implement something called "sticky sessions" across many servers, that means you can keep a user's login even if he connects to a different server of the same website. Redis provides, stores random data that needs to be accessed fast by many servers, implementing a message or job queue.

3. **Why do we need redis if we have databases?**

Ans 3.

Redis enables you to write traditionally complex code with fewer, simpler lines. With Redis, you write fewer lines of code to store, access, and use data in your applications. The difference is that developers who use Redis can use a simple command structure as opposed to the query languages of traditional databases.

a. Redis advantages and disadvantages.

Advantages of Redis:

1) **It's super-fast. Faster than any other cashing out there.**
2) **Due to easy setup, Redis is Simple and easy to use.**
3) **Redis has flexible data structures, it supports almost all data structures.**
4) **Redis allows storing key and value pairs as large as 512 MB.**
5) **Redis uses its own hashing mechanism called Redis Hashing.**
6) **Zero downtime or performance impact while scaling up or down.**
7) **Last and probably the very obvious point, it is open source and stable**

Disadvantages Of Redis:

1) **Data is sharded based on the hash-slots assigned to each Master. If the Master holding some slots is down, data to be written to that slot will be lost.**
2) **Clients connecting to the Redis cluster should be aware of the cluster topology, causing overhead configuration on Clients.**
3) **Failover does not happen unless the master has at least one slave.**
4) **It requires a huge ram because it is in-memory so you are not supposed to use it on ram servers.**

4. Explain cache miss and cache hit.

Ans 4.

A cache miss is an event in which a system or application makes a request to retrieve data from a cache, but that specific data is not currently in cache memory. Contrast this to a cache hit, in which the requested data is successfully retrieved from the cache.

Other (OOPS + Git + Multithreading)

1. OOPS overview.

Ans 1.

Object-oriented programming (OOP) is a computer programming model that organizes software design around data, or objects, rather than functions and logic.

2. What is inheritance and abstraction?

Ans 2.

Inheritance :- A mechanism where you can derive a class from another class for a hierarchy of classes that share a set of attributes and methods.

Abstraction :- A programmer hides all but the relevant data about an object in order to reduce complexity and increase efficiency.

3. **What is Git?**

Ans 3.

Git is a tool used for source code management. Git is used for tracking changes in the source code, enabling multiple developers to work together.

4. What are some basic git commands?

Ans 4.

- **Git clone**
- **Git checkout**
- **Git status**
- **Git add**
- **Git commit**
- **Git push**
- **Git pull**

5. **What is a Git repository and branches?**

Ans 5.

A repository is your whole project (directories and files) that you clone on your computer. A branch is a version of your repository

6. What is difference between Git and GitHub

Ans 6.

S.No.	Git	GitHub
1.	Git is a software.	GitHub is a service.
2.	Git is a command-line tool	GitHub is a graphical user interface
3.	Git is installed locally on the system	GitHub is hosted on the web
4.	Git is maintained by linux.	GitHub is maintained by Microsoft.
5.	Git is focused on version control and code sharing.	GitHub is focused on centralized source code hosting.
6.	Git is a version control system to manage source code history.	GitHub is a hosting service for Git repositories.

S.No.	Git	GitHub
7.	Git was first released in 2005.	GitHub was launched in 2008.
8.	Git has no user management feature.	GitHub has a built-in user management feature.
9.	Git is open-source licensed.	GitHub includes a free-tier and pay-for-use tier.
10.	Git has minimal external tool configuration.	GitHub has an active marketplace for tool integration.
11.	Git provides a Desktop interface named Git Gui.	GitHub provides a Desktop interface named GitHub Desktop.
12.	Git competes with CVS, Azure DevOps Server, Subversion, Mercurial, etc.	GitHub competes with GitLab, Bitbucket, AWS Code Commit, etc.

7. What is class,id in html? What is DOM?

Ans 7.

ID and class are two of the mostly used CSS selectors that not only help with building the layout of the HTML document but they also help with styling it.

The Document Object Model is a cross-platform and language-independent interface that treats an XML or HTML document as a tree structure wherein each node is an object representing a part of the document.

Mongodb

1. Why is a database? Name a few examples.

Ans 1.

A database is a collection of information or data which is organized in such a way that it can be easily accessed, managed and retrieved.

2. Why you use MongoDB over MySQL and PostgreSQL? MySql vs MongoDb (or SQL/RDBMS vs NoSql).

Ans 2.

While NoSQL databases work on storing data in key-value pairs as one record, relational databases store data on different tables. If you prioritize faster data integration and scalability across several servers, MongoDB might be a suitable choice for your business.

	SQL	NoSQL
Database Type	Relational Databases	Non-relational Databases / Distributed Databases
Structure	Table-based	• Key-value pairs • Document-based • Graph databases • Wide-column stores
Scalability	Designed for scaling up vertically by upgrading one expensive custom-built hardware	Designed for scaling out horizontally by using shards to distribute load across multiple commodity (inexpensive) hardware
Strength	• Great for highly structured data and don't anticipate changes to the database structure • Working with complex queries and reports	• Pairs well with fast paced, agile development teams • Data consistency and integrity is not top priority • Expecting high transaction load

3. Compare relational and non-Relational databases and their benefits or drawbacks?
Ans 3.

RELATIONAL DATABASE	NONRELATIONAL DATABASE
A database based on the relational model of the data, as proposed by E.F. Codd in 1970	A type of database that provides a mechanism for storing and retrieving data that is modeled in a way other than the tabular relations used in relational databases
Also called SQL databases	Also called NoSQL databases
Tables can be joined together	There is no joint concept
Use SQL	Do not use SQL
Cannot be categorized further	Types include key-value, documents, column, and graph databases
Help to achieve complex querying, provide flexibility and help to analyze data	Work well with a large amount of data, reduce latency and improve throughput
Ex: MySQL, SQLite3, and, PostgreSQL	Ex: Cassendra, Hbase, MongoDB, and, Neo4

4. What kind of db is MongoDb? Why do we define schema in nodejs application when working with Mongodb if it is schema-less?
Ans 4.
MongoDB is a non-relational document database that provides support for JSON-like storage. The MongoDB database has a flexible data model that enables you to store unstructured data, and it provides full indexing support, and replication with rich and intuitive APIs.

You need to know the names, types and meanings of those fields. That's a schema. When people say that MongoDB "has no schema", they really mean that it does not enforce schema the way SQL databases do. MongoDB pushes schema concerns up to your application level, where you can handle them more flexibly.

5. What is an aggregation pipeline? What are some stages used in it?
Ans 5.

An aggregation pipeline consists of one or more stages that process documents: Each stage performs an operation on the input documents.
These are 7 stages of aggregation of Pipeline in MongoDB:

1. **$project**
2. **$match**
3. **$group**
4. **$sort**

5. $skip & $limit
6. $first & $last
7. $unwind

6. What is a Schema and a model in mongoose? How are they different?
Ans 6.

A Mongoose schema defines the structure of the document, default values, validators, etc., whereas a Mongoose model provides an interface to the database for creating, querying, updating, deleting records, etc.

7. Discuss some common mongoose APIs for CRUD operations.
Ans 7.

The basic methods of interacting with a MongoDB server are called CRUD operations. CRUD stands for Create, Read, Update, and Delete. These CRUD methods are the primary ways you will manage the data in your databases.

8. Why do you have to use await if you want to process the result.
Ans 8.

The await keyword before a promise makes JavaScript wait until that promise settles, and then: If it's an error, an exception is generated — same as if a throw error were called at that very place. Otherwise, it returns the result.

9. What are indexes? What is the default index value set by MongoDB?
   a. **What do you mean by database indexing?**
Ans 9.

Default Index. In MongoDB indexing, all the collections have a default index on the _id field. If we don't specify any value for the _id the MongoDB will create _id field with an object value.

A database index is a data structure that improves the speed of data retrieval operations on a database table at the cost of additional writes and storage space to maintain the index data structure.

10. How to link the document of one collection to a document in another collection using mongoose. Explain reference in mongoose.
Ans 10.

We can link one collection of a document to another collection of documents using reference and populate, which means providing a reference to one document to another document and then using populate we can show that reference data.

DB Refs are references from one document to another using the value of the first document's _id field, collection name, and, optionally, its database name, as well as any other fields. DBRefs allow you to more easily reference documents stored in multiple collections or databases.

11. What is the difference between find, findOne, findOneAndUpdate and findById APIs.
Ans 11.

The main difference between these four methods is how MongoDB handles the condition given to them Like in find() it gives the whole document in findOne() it gives only one document related to condition in findById() it finds the doc through particular id that is assign to them while saving in database and in findOneAndUpdate() finds a doc and updates according to the condition

12. Also between updateOne, update and updateMany? When should you use each?
Ans 12.

updateMany: updates all the documents that match the filter. updateOne: updates only one document that match the filter. And update is the deprecated instead of this we use updateOne() and updateMany()

API

1. What is an API?
Ans 1.

APIs are mechanisms that enable two software components to communicate with each other using a set of definitions and protocols. For example, the weather bureau's software system contains daily weather data.

2. What are rest api? Why are they stateless?
Ans 2.

A RESTful API is an architectural style for an application program interface (API) that uses HTTP requests to access and use data.
REST APIs are stateless because, rather than relying on the server remembering previous requests, REST applications require each request to contain all of the information necessary for the server to understand it.

3. What are the different ways to pass data to an API while requesting a Create operation? Discuss the Content-Type header in brief
Ans 3.

REST API endpoints can pass data within their requests through 4 types of parameters: Header, Path, Query String, or in the Request Body.
The Content-Type representation header is used to indicate the original media type of the resource (prior to any content encoding applied for sending).
SYNTAX -
Content-Type: text/html; charset=utf-8
Content-Type: multipart/form-data; boundary=something

4. Difference between Put and Patch
Ans 4.

PUT is a method of modifying a resource where the client sends data that updates the entire resource .
PATCH is a method of modifying resources where the client sends partial data that is to be updated without modifying the entire data.

5. What is an HTTP status code? Discuss the popularly used codes along with their use case.
Ans 5.
HTTP response status codes indicate whether a specific HTTP request has been successfully completed. Responses are grouped in five classes:

The most important status codes for SEOs
HTTP Status Code 200 - OK. ...
HTTP Status Code 201 - Created. ...
HTTP Status Code 301 - Permanent Redirect. ...
HTTP Status Code 400 – Bad Request. ...
HTTP Status Code 401 - Unauthorized. ...
HTTP Status Code 404 - Not Found. ...
HTTP Status Code 500 - Internal Server Error. ...

6. What are http status codes? Please elaborate?
Ans 6.

The Status-Code element in a server response, is a 3-digit integer where the first digit of the Status-Code defines the class of response and the last two digits do not have any categorization role. There are 5 values for the first digit:

- **Informational responses (100 – 199)**
- **Successful responses (200 – 299)**
- **Redirection messages (300 – 399)**
- **Client error responses (400 – 499)**
- **Server error responses (500 – 599)**