What is Node.js? V.IMP.

Node js was developed by Rayn dahl in 2009. Node is neither a language nor a framework. Node/ Node.js a runtime environment for executing JavaScript code on the server side. Also it allows developers to use same Language in frontend and backend. Nodejs allows to run the JavaScript code outside the browser.

How Node is a runtime environment on server side? What is V8?

Browsers execute JavaScript on the client side, and similarly, Node.js executes JavaScript on the server side. Nodejs is built on the top of v8 Engine. V8 Engine is a JavaScript Engine which is used by the chrome browser and developed by the google which is known for its high performance that makes Js code execution efficient.

Runtime Environment: - It is that where the program will be executed.

What is REPL In Node JS?

- READ – This reads the input provided by the user, parses it into JavaScript data structure, and stores it in the memory.
- EVAL – This executes the data structure.
- PRINT – This prints the outcome generated after evaluating the command.
- LOOP – This loops the above command until the user presses Ctrl+C twice.

What is the difference between Runtime environment & Framework?

Runtime Environment: Primarily focuses on providing the necessary infrastructure for code execution, including services like memory management and I/O operations.

Framework: Primarily focuses on simplifying the development process by providing built in support features.

Nodejs and JavaScript are same?

No, JavaScript and Nodejs is completely different things. JavaScript is a language which is primarily used in client side while Nodejs is a runtime environment that allows you to use JavaScript on server side.

What is the difference between Node.js & Express.js?

Node.js is a runtime environment that allows the execution of JavaScript code server-side.
Express.js is a framework built on the top Node.js It is designed to simplify the process of building web applications and APIs by providing a set of features like simple routing system, middleware support etc.

What are the differences between Client Side(Browser) & Server-Side(Node.js)?

Client-Side (Browser)

Runs on the user's web browser. HTML, CSS, JavaScript. Handles UI display, interactions, and client-side logic.

Server-Side (Node.js/server)

Runs on the server. Handles business logic, data storage access, authentication, authorization etc.

What are the 7 Main Features of Node.js? V.IMP.

1. Single Threaded 2. Asynchronous 3. Event-Driven 4. V8 JavaScript Engine 5. Cross-Platform 6. NPM (Node package Manager) 7. Real-Time Capabilities

What is Synchronous Programming?

In a synchronous program, each task is performed one after the other, and the program waits for each operation to complete before moving on to the next one.

What is Multi-Threaded Programming?

In a multi-threaded application, multiple threads can execute multiple tasks simultaneously and parallelly. As soon as one thread starts a task, the next task is immediately initiated by creating a second thread.

What is Asynchronous Programming? V. IMP.

In Node.js, if there are 4 tasks (Task1, Task2, Task3, Task4) to be completed for an event. Then below steps will be executed: First, Thread T1 will be created. Thread T1 initiates Task1, but it won't wait for Task1 to complete. Instead, T1 proceeds to initiate Task2, then Task3 and Task4(This asynchronous execution allows T1 to efficiently handle multiple tasks concurrently). Whenever Task1 completes, an event is emitted. Thread T1, being event-driven, promptly responds to this event, interrupting its current task and delivering the result of Task 1

what is the difference between Synchronous & Asynchronous programming?

Synchronous programming	Asynchronous programming
1. In synchronous programming, tasks are executed one after another in a **sequential manner**.	In synchronous programming, tasks can start, run, and complete in parallel
2. Each task must complete before the program moves on to the next task.	Tasks can be executed independently of each other.
3. Execution of code is blocked until a task is finished.	Asynchronous operations are typically non-blocking.
4. Synchronous operations can lead to blocking and unresponsiveness.	It enables better concurrency and responsiveness.

How non-blocking operation is performed by Event loop?

Non-blocking requests are a form of asynchronous programming where the system does not wait for a request to complete before moving on to other tasks. Event loops process these requests and send responses directly to the client once they are ready.

How Blocking operation is performed by Event loop?

Blocking request is synchronous programming which execute line by line means it does not execute the next task until the previous task is not completed.

What are Events, Event Emitter, Event Queue, Event Loop & Event Driven?

Event: Signals that something has happened in a program.

Event Emitter: Create or emit events.

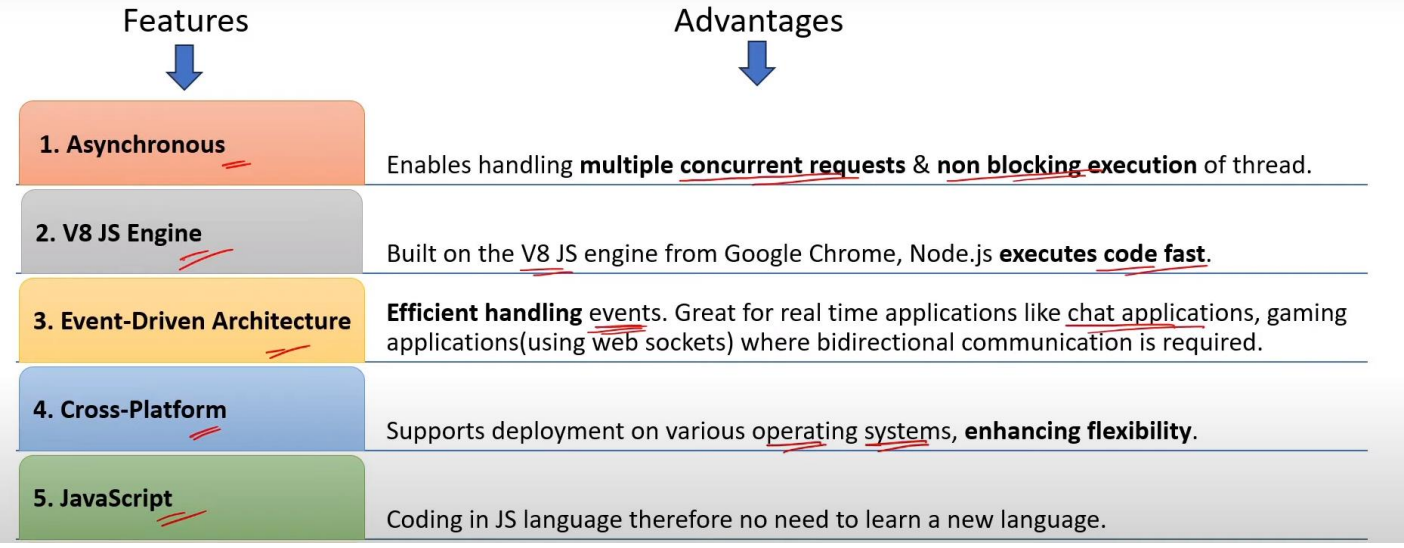
Event Queue: The Events which are emitted are stored in event queue.

Event Handler (Event Listener): Function that responds to specific events.

Event Loop: The event loop picks up event from the event queue and executes them in the order they were added.

Event Driven Architecture: It means operations in Node are drive or based by events.

What are the main features & advantages of Node.js?



What are the disadvantages of node? When to use and when not to use Node?

Ideal for real-time applications like chat applications, online gaming, and collaborative tools due to its event-driven architecture.

Excellent for building lightweight and scalable RESTful APIs that handle a large number of concurrent connections.

Well-suited for building microservices-based architectures, enabling modular and scalable systems.

Disadvantage CPU-Intensive Tasks: Avoid for applications that involve heavy CPU processing (Image/Video Processing, Data Encryption/Decryption) as Node.js may not provide optimal performance in such scenarios because it single threaded and for heavy computation multi-threaded is better.

What is RESTful API?

A RESTful API (Representational State Transfer API) is an architectural style for building web APIs. It utilizes HTTP methods like GET, POST, PUT, and DELETE to execute CRUD (create, read, update, delete) operations on resources, identifying these resources through URLs. A key characteristic of a RESTful API is its statelessness, signifying that each request carries all the essential information for its completion.

What is NPM? What is the role of node_modules folder?

NPM (Node Package Manager) is used to manage the dependencies for your Node project.

node_modules folder contains all the dependencies of the node project.

What is the role of package.json file in Node?

The package.json file contains project metadata (information about the project). For example, project name, version, description, author, license etc.

What are Modules in Node? What is the difference between a function & module?

A module contains a specific functionality that can be easily reused within a Node.js application. Modules can contain multiple functions and variables.

What is nodemon?

It is a package which is designed to monitor the Nodejs application file and automatically restart the server when any changes are detected.

Name some of the widely used Node.js libraries?

Express JS - It is a web application framework that provides a wide set of features that make web application development easier and faster.

Mongoose - It is an object modeling tool that is used to create a connection from the application to the database.

Q. How many ways are there to Export a module?

1. **module.exports** object can be used to export the module.

```
// Exporting a function
function sayHello(name) {
  console.log("Hello, " + name);
}

// Exporting a variable
const pi = 3.14159;

// Exporting an object
const myObject = {
  key: "value",
};

// Using module.exports
module.exports.sayHello = sayHello;
module.exports.pi = pi;
module.exports.myObject = myObject;
```

2. **exports** object can be also used directly to export the module.

```
// Exporting a function
exports.sayHello = function (name) {
  console.log("Hello, " + name);
};

// Exporting a variable
exports.pi = 3.14159;

// Exporting an object
exports.myObject = {
  key: "value",
};
```

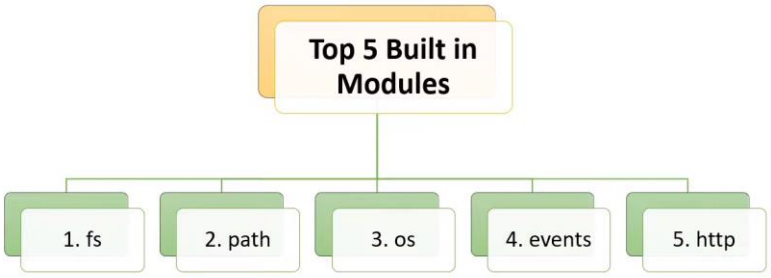
What is module wrapper function?

In Node.js, each module is wrapped in a function called the "module wrapper function" before it is executed.

Q. What are the Types of modules in Node?

1. Built-in Module (Core Modules)	These are **already present modules** in Node.js which provide essential functionalities. For example, fs (file system), http (HTTP server), path (path manipulation), and util (utilities).
2. Local Modules	These **user-defined modules(JS files)** created by developers for specific functionalities.
3. Third-Party Modules	These are **external packages or libraries created** by the community and provide additional functionalities for Node projects. You install third-party modules using the npm install command. For example, npm install lodash.

Q. What are the Top 5 built in modules commonly used in node projects?



Q. Explain the role of fs module? Name some functions of it?

- fs (File System) module in Node provides a set of methods for **interacting with the file system**.

```
// fs-example.js
const fs = require("fs");

// Reading the contents of a file asynchronously
fs.readFile("fs.txt", "utf8", (err, data) => {
  if (err) {
    return;
  }
  console.log("File contents:", data);
});

// Writing to a file asynchronously
const contentToWrite = "Some content";
fs.writeFile("fs.txt", contentToWrite, "utf8", (err) =>
  if (err) {
    return;
  }
  console.log("Write operation complete.");
});
```

Q. Explain the role of fs module? Name some functions of it?

- **7 Main functions of fs module:**

1. fs.readFile()	Reads the content of the file specified
2. fs.writeFile()	Writes data to the specified file, creating the file if it does not exist.
3. fs.appendFile()	Appends data to a file. If the file does not exist, it is created.
4. fs.unlink()	Deletes the specified file.
5. fs.readdir()	Reads the contents of a directory.
6. fs.mkdir()	Creates a new directory.
7. fs.rmdir()	Removes the specified directory.

Q. Explain the role of path module? Name some functions of it?

- **5 Main functions of path module:**

```
const path = require('path');

// Joining path segments together
const fullPath = path.join(__dirname, 'folder', 'file.txt');

// Resolving the absolute path
const absolutePath = path.resolve('folder', 'file.txt');

// Getting the directory name of a path
const directoryName = path.dirname('/folder/file.txt');

// Getting the file extension of a path
const fileExtension = path.extname('/folder/file.txt');

// Parsing a path into an object with its components
const pathObject = path.parse('/folder/file.txt');
```

Path module provides utilities for joining, resolving, parsing, formatting, normalizing, and manipulating paths.

Q. Explain the role of OS module? Name some functions of it?

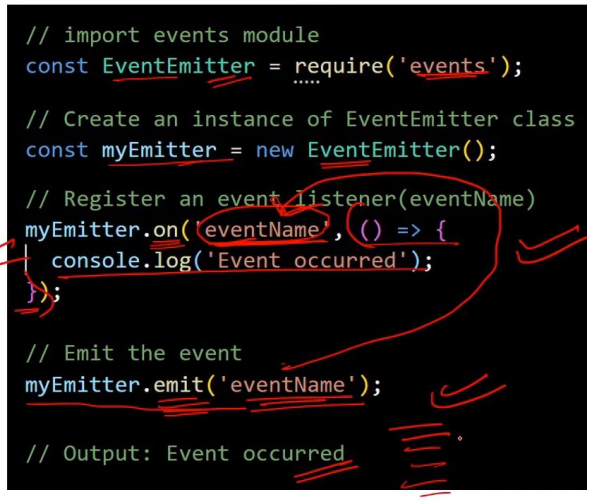
- The os module in Node.js provides a set of methods for **interacting with the operating system**.
- Operation system can be used by developers for building cross-platform applications or performing system-related tasks in Node.js applications.

```
const os = require('os');

// 1. Fet Platform Information
console.log(os.type());
// Output: 'Windows_NT' or 'Linux'...

// 2. Get Current User Information
console.log(os.userInfo());
// Output: {uid: -1, gid: -1, username: 'anaya'..

// 3. Get Memory Information in bytes
console.log(os.totalmem()); // Output: 1487726591
console.log(os.freemem()); // Output: 4961570816
```

Cross platform

Q. Explain the role of events module? How to handle events in Node?

1. **events module** is used to handle events.
2. **EventEmitter** class of events module is used to register event listeners and emit events.
3. An **event listener** is a function that will be executed when a particular event occurs.
4. **on()** method is used to register event listeners.



```
// import events module
const EventEmitter = require('events');

// Create an instance of EventEmitter class
const myEmitter = new EventEmitter();

// Register an event listener(eventName)
myEmitter.on('eventName', () => {
  console.log('Event occurred');
});

// Emit the event
myEmitter.emit('eventName');

// Output: Event occurred
```

What is the role of http module in node? v.imp.

The HTTP module can create an HTTP server that listens to server ports and gives a response back to the client.

What is the role createServer() method of http module?

The createServer() method of the http module in Node.js is used to create an HTTP server.

Express.js

Express.js is a small framework that works on top of Node.js web server functionality to simplify its APIs and add helpful new features. It makes it easier to organize your application's functionality with middleware and routing. It adds helpful utilities to Node.js HTTP objects and facilitates the rendering of dynamic HTTP objects.

Why use Express.Js?

Express.js is a lightweight Node.js framework that gives us ability to create server-side web applications faster and smarter. The main reason for choosing Express is its simplicity, minimalism, flexibility, and scalability characteristics. It provides easy setup for middlewares and routing.

Differentiate between Node.js and Express.js?

Node.js is the runtime environment that allows you to execute JavaScript on the server side, on the other hand Express.js is a framework built on top of Node.js that provides a set of tools for building web applications and APIs.

Express.js is not the only framework available for Node.js, but it is widely used because to its simplicity and flexibility.

Is Express JS a front-end or a back-end framework?

Express.js is a JavaScript backend framework. It is mainly designed to develop complete web applications and APIs. Express is the backend component of the MERN stack which stands for MongoDB, Express.js, React.js, Node.js.

Which major tools can be integrated with Express JS?

There are many tools and libraries that can be integrated with Express.js such as:

- Database tools: MongoDB, MySQL, PostgreSQL.
- Template Engines: EJS, Pug, Mustache.
- Authentication libraries: Passport.js.

What is the Template Engine?

Template Engine enables you to use static template file in your application. At runtime, Template engine replace a variable in the static template file with actual values and convert template into html file sent to the client. This Approach Make it easier the design of html page. It simple terms, we can say that create a dynamic HTML page with the help of template engine.

What is Ejs?

Ejs is a Embedded JavaScript that is a Template Engine is used to generate Html page with plain JavaScript. It also helps to embed JavaScript on the html page.

What is .env file used for?

dotenv is a tool and library used for storing sensitive information in a web application which we don't want to expose to others like password, database connection string etc.

What are JWT?

Json Web Tokens are mainly a token which is used for authentication and information exchange. When a user signs in to an application, the application then assigns JWT to that user.

What is Bcrypt used for?

Bcrypt is a password hashing function which is used to securely hash and store user passwords.

Differentiate between res.send() and res.json().

Both res.send() and res.json() serves similar purposes with some difference. So it depends on the data type which we are working with. Choose res.json() when you are specifically working with JSON data.

Explain what CORS is in Express JS?

CORS (Cross-Origin Resource Sharing) is a security feature implemented by web browsers to control how web pages in one domain can request and interact with resources hosted on another domain.

In the context of Express.js, CORS refers to a middleware that enables Cross-Origin Resource Sharing for your application. This allows the application to control which domains can access your resources by setting HTTP headers.

Name some databases that integrate with Express JS?

Express.js can support a variety of the databases which includes:

- MySQL
- MongoDB
- PostgreSQL
- SQLite
- Oracle

What are middlewares in Express.Js?

Middleware functions are those functions that have the access to request and response object and the next middleware or function. They can add functionality to an application, such as logging, authentication, and error handling.

What is body parser?

body parser is middleware for express.js. it extract the data of coming from HTTP request, converts it from JSON format into a JavaScript object, and put inside the req.body.

What is Express router() function?

The express.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.

How to handle routing in Express JS?

Express.js manages routing through the use of the `express.Router()` method.

What are the different types of HTTP requests?

The primary HTTP methods are commonly referred to as CRUD operations, representing Create, Read, Update, and Delete. Here are the main HTTP methods:

- GET: The GET method is used to request data from a specified resource.
- POST: The POST method is used to submit data to be processed to a specified resource.
- PUT: The PUT method is used to update a resource or create a new resource if it does not exist.
- PATCH: The PATCH method is used to apply partial modifications to a resource.
- DELETE: The DELETE method is used to request that a specified resource be removed.

Which are the arguments available to an Express JS route handler function?

In Express JS route handler function, there are mainly3 arguments available that provide useful information and functionality.

- req: This represents the HTTP request object which holds information about the incoming request. It allows you to access and manipulate the request data.
- res: This represents the HTTP response object which is used to send the response back to the client. It provides methods and properties to set response headers, status codes, and send the response body.
- next: The next() function is used to pass control from one middleware function to the next function. It is used to execute the next middleware function in the chain.

How can you deal with error handling in Express.js?

Express.js provides built-in error-handling mechanism with the help of the next() function. When an error occurs, you can pass it to the next middleware or route handler using the next() function. You can also add an error-handling middleware to your application that will be executed whenever an error occurs.

What is the purpose of the next() function in Express.js?

The next() function is used to pass control from one middleware function to the next function. It is used to execute the next middleware function in the chain.

MongoDB

What is MongoDB ?

MongoDB is an open-source NoSQL database written in C++ language. . It uses JSON-like documents with dynamic schemas.. It is designed for scalability, high performance and it is suitable for applications which requires large-scale data storage.

MongoDB works on the concept of Collection and Document.

MongoDB is developed by MongoDB Inc. in 2009.

What are some of the advantages of MongoDB?

Some advantages of MongoDB are as follows:

- Flexible Schema: It uses JSON-like documents with dynamic schemas..
- High Performance: Optimized for read and write operations It is designed for high performance.
- Scalability: it is suitable for applications which requires large-scale data storage.

what is difference sql and nosql?

SQL databases use structured schemas and store data in tables with rows and columns, ideal for complex queries and transactions. NoSQL databases use flexible schemas and store data in various formats like documents, key-value pairs, and graphs, suited for scalable and dynamic data handling.

What are the data types in MongoDB?

The prevalent data types in MongoDB include:

- Null
- Boolean
- Number
- String
- Date
- Array
- object

what is schema and models?

Mongoose allows you to create a schema for your document which represent the structure and datatype of your document within a collection.

Schema is representation of specified collection in mongoDB database like a person.

What is a Document in MongoDB?

In MongoDB, a document is a record in a collection, represented in a JSON-like format (BSON). It consists of field and value pairs, allowing for flexible and hierarchical data structures, making it easy to store complex data types and arrays.

What is a Collection in MongoDB?

In MongoDB, a collection is a group of documents, similar to a table in a relational database. Collections hold multiple documents and do not enforce a schema, allowing for flexible and dynamic data storage.

What are Databases in MongoDB?

MongoDB groups collections into databases. MongoDB can host several databases, each grouping together collections.

Some reserved database names are as follows:

admin

local

config

What is the Mongo Shell?

It is a JavaScript shell that allows interaction with a MongoDB instance from the command line. With that one can perform administrative functions, inspecting an instance, or exploring MongoDB.

What Is Replication In MongoDB?

Creating multiple copies of data across different servers, known as replica sets.

What is the purpose of MongoDB?

MongoDB serves as a document-oriented database manager specifically crafted for the storage of substantial data volumes. It stores data in a binary JSON format and incorporates the concepts of collections and documents. Being a cross-platform, NoSQL database, MongoDB is distinguished by its high performance, scalability, and flexibility, enabling smooth querying and indexing operations.

Explain the difference between a document and a collection in MongoDB.

A document is a single record in MongoDB, similar to a row in SQL, and a collection is a group of documents, like a table in SQL.

How do you create a new database in MongoDB?

A new database is created automatically when you insert data into it.

Explain the concept of sharding in MongoDB.

Sharding is a method of distributing data across multiple servers to handle large datasets.

How do you perform a query to find all documents in a collection where a specific field matches a given value?

Use the find method, e.g., db.collection.find({ fieldName: value }).

What is an index in MongoDB and why is it important?

An index improves the speed of search queries by organizing data efficiently.

How do you insert a document into a MongoDB collection?

Use the insertOne method, e.g., db.collection.insertOne({ field1: value1, field2: value2 }).

Explain the concept of a schema in the context of MongoDB.

MongoDB is schema-less, meaning documents in the same collection can have different structures.

What is the purpose of the ObjectId in MongoDB?

ObjectId is a unique identifier for each document in a MongoDB collection.