## **What is node.js? How node.js works?**

Node.js is an open-source, cross-platform runtime environment for developing server-side applications written in JavaScript. Node.js allows us to run JavaScript on the server. JS is designed to use single-threaded, event-based architecture and non-blocking-based I/O models. It eliminates the waiting, and simply continues with the next request.

It is made of Google Chrome’s V8 engine written in multiple-platform C library i.e. C++ and Libuv. Node JS Platform does not follow Request/Response Multi-Threaded Stateless Model. It follows Single Threaded with Event Loop Model. Node JS Processing model mainly based on Javascript Event based model with Javascript callback mechanism. So the server never needs to create additional threads or switch between threads, which means it has very little overhead.

Basically, the server consists of one thread processing one event after another. Node is completely event-driven. The server starts processing event and when there is a blocking IO operation, it does not wait until it completes and instead registers a callback function. The server then immediately starts to process another event (maybe another request). When the IO operation is finished, that is another kind of event, and the server will process it (i.e. continue working on the request) by executing the callback as soon as it has time.

Below mentioned are some advantages of using Node.js

* Easy to learn because of its accessibility and object-oriented programming.
* Simple to use because it only uses one language as the catalyst of the full-stack development.
* It is Faster to deploy, as programs and websites built with Node.js require less time on development hence reducing the overall time-to-market.

## **Why is Node.js Single threaded?**

Node.js is a single threaded application with event looping for async processing. The biggest advantage of doing async processing on a single thread is, can handle more concurrent client’s requests with ease. Eliminates the need of creating more threads, because of the Event loop.

## **What is npm? What is the main functionality of npm?**

npm stands for Node Package Manager. Following are the two main functionalities of npm:

* Online repositories for node.js packages/modules.
* Command line utility to install packages, do version management and dependency management of Node.js packages.

## **Explain the below**

**Middleware** is a function that has the access to the Request, Response, Next() in the application’s request-response cycle. ‘next’ is a parameter which is passed in middleware function to pass control to next middleware. Some Tasks: Executed any type of code; Update the request and the response objects; Finish the request-response cycle; invoke the next middleware in the stack;

**Control Flow** A generic piece of code which runs in between several asynchronous function calls is known as control flow function.

**First Class Functions**: When functions can be treated like any other variable then those functions are first-class functions.

**(Types) API Functions** has 2 types: Asynchronous, Non-blocking functions & Synchronous, Blocking functions

**Call-back** function is called after a given task. It allows other code to be run in the meantime and prevents any blocking.  Being an asynchronous platform, Node.js heavily relies on callback. All APIs of Node are written to support callbacks.

**EventEmitter** is a Node.js class that includes all the objects that are basically capable of emitting events. This can be done by attaching named events that are emitted by the object using an eventEmitter.on() function. Thus whenever this object throws an even the attached functions are invoked synchronously.

**REPL** stands for Read Eval Print Loop. It specifies a computer environment like a window console where we can enter a command, and the computer responds with an output. It is very useful in writing and debugging the codes. REPL environment incorporates Node.js.

**I/O** is the shorthand for input and output, and it will access anything outside of your application. It will be loaded into the machine memory to run the program once the application is started.

**Event-Driven** programming is a programming paradigm in which the flow of the program is determined by events like messages from other programs or threads. It is an application architecture technique divided into two sections 1) Event Selection 2) Event Handling.

## **What is the difference between fork() and spawn() methods in Node.js?**

Spawn command is designed to run system commands. When Spawn command is used, you send it a system command that is processed on its own but no further code is processed within your node process. Whereas, Fork is a special instance of Spawn. This means that you can create workers for running the exact same Node code base or maybe a different module for a particular task.

## **What is the buffer class in Node.js?**

Buffer class stores raw data similar to an array of integers but corresponds to a raw memory allocation outside the V8 heap. Buffer class is used because pure JavaScript is not compatible with binary data

## **What is node.js streams?**

Streams are instances of EventEmitter which can be used to work with streaming data in Node.js. They can be used for handling and manipulating streaming large files (videos, mp3, etc) over the network. They use buffers as their temporary storage. There are four types:

* Writable: streams to which data can be written (for example, fs.createWriteStream()).
* Readable: streams from which data can be read (for example, fs.createReadStream()).
* Duplex: streams that are both Readable and Writable (for example, net.Socket).
* Transform: Duplex streams that can modify or transform the data as it is written and read (for example, zlib.createDeflate()).

## **Explain what a Reactor Pattern in Node.js?**

Reactor pattern again a pattern for nonblocking I/O operations. But in general, this is used in any event-driven architecture.   
There are two components in this: 1. Reactor 2. Handler.  
  
**Reactor**: Its job is to dispatch the I/O event to appropriate handlers  
**Handler**: Its job is to actually work on those events

--------------------------------xx-----------------------------------xx---------------------------------

## **What is ExpressJs and its features ?**

Express Js is a framework for node.js which is light-weight and fast. It is used to develop web and mobile applications. Following are the features of Expressjs:

* It allows middleware set up to respond HTTP Requests
* It defines routing table which is used to achieve action based on HTTP Method and URL.
* It allows dynamically generate the HTML Pages based on passing arguments to the templates.
* It follows MVC architecture for web application.

## **What is middleware in ExpressJs?**

Middleware is a function that accesses the request object (req), response object (res), and access next middleware function in application’s request-response cycle. ‘next’ is a parameter which is passed in middleware function to pass control to next middleware. **Syntax:** function(req,res,next){ }

Middleware is able to handle error handling in ExpressJs. This can be done by passing one extra error-handling (err) parameter in middleware function.

**Syntax:** function(err,req,res,next){ }

## **What is routing?**

Routing define how an application responds over client request to a particular endpoint (URI). Endpoint is specially a path and any one of HTTP request methods (GET, POST, etc).

Following structure is used for routing:

app.METHOD(PATH,HANDLER)

Note: METHOD is Http request method (in lowercase),

PATH is uri on server,

HANDLER is function name.

## **What is scaffolding?**

Scaffolding is a tool, which set up all required public directory, add middleware, create separate route file etc (set up skeleton for web application). So that we can directly get started building our application.  
**Yeoman** is scaffolding tool built for Node.js.

## **How to Config properties in Express JS?**

In Express JS, there are 2 ways that for configuring the properties:

With process.ENV:

* A file with the name “.env” is to be created within the project folder.
* All the properties are to be other within the “.env” file.
* Any of the properties will be employed in server.js.

With require.JS:

* A file with the name “config.json” is to be created within the config folder within the project folder.
* The config properties are to be present within the config.json file.
* Now, ought to be accustomed access the config.json file.

-------------------------xx----------------------------xx------------------------------

## **Explain what is MongoDB, Namespace & Sharding ?**

Mongo-DB is a document database which provides high performance, high availability and easy scalability. MongoDB stores BSON (Binary Interchange and Structure Object Notation) objects in the collection. The concatenation of the collection name and database name is called a namespace. The procedure of storing data records across multiple machines is referred as Sharding. It is a MongoDB approach to meet the demands of data growth. It is the horizontal partition of data in a database or search engine. Each partition is referred as shard or database shard.

## **Explain what is a replica set and how replication works in MongoDB?**

A replica set is a group of mongo instances that host the same data set. In replica set, one node is primary, and another is secondary. From primary to the secondary node all data replicates.

Across multiple servers, the process of synchronizing data is known as replication. It provides redundancy and increase data availability with multiple copies of data on different database server. Replication helps in protecting the database from the loss of a single server.

## **What is a Document, Collection, Databases & Mongo Shell in MongoDB?**

**A Document** in MongoDB is an ordered set of keys with associated values. It is represented by a map, hash, or dictionary. In JavaScript, documents are represented as objects:

{"greeting" : "Hello world!"}

Complex documents will contain multiple key/value pairs:

{"greeting" : "Hello world!", "views" : 3}

**A collection** in MongoDB is a group of documents. If a document is the MongoDB analog of a row in a relational database, then a collection can be thought of as the analog to a table.

Documents within a single collection can have any number of different “shapes.”, i.e. collections have dynamic schemas.

For example, both of the following documents could be stored in a single collection:

{"greeting" : "Hello world!", "views": 3}

{"signoff": "Good bye"}

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

Some reserved database names are as follows:

* admin
* local
* config

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.

To start the shell, run the mongo executable:

$ mongod

$ mongo

MongoDB shell version: 4.2.0

connecting to: test

>

The shell is a full-featured JavaScript interpreter, capable of running arbitrary JavaScript programs. Let’s see how basic math works on this:

> x = 100;

200

> x / 5;

20

-------------------------xx-----------------------------xx------------------------------

## **What is JIRA?**

JIRA is developed by Atlassian and it is a software tool used for bug tracking, project management, and issue tracking. It is widely used in software development and software testing.

* Able to track project progress from time to time.
* JIRA use-cases include project management, feature implementation, bug tracking, etc.
* Workflow can be easily customized as per our requirement.
* Along with issue tracking, history of the work done on issues, when, what and by whom can also be tracked.
* JIRA is platform-independent and can run anywhere.

## **Explain the JIRA workflow.**

Workflow defines the series of steps an issue goes through during its lifecycle from creation to closing of the issue.

## **Explain the three color indicators and their significance.**

For any issue in JIRA, 3 colors like Blue, Green, and Orange is used to denote the amount of time spent on issue. Each color has its own significance.

* Blue (Estimated): time estimate to be invested in resolving the issue.
* Orange (Remaining): time left for resolving the issue.
* Green (logged): time that has been used in resolving the issue so far.

## **What is Postman?**

Postman is a collaboration platform for API development. Using the Postman tool, we can send HTTP/s requests to a service, as well as get their responses. By doing this we can make sure that the service is up and running.

## **What is an API?**

API stands for Application Programming Interface. API acts as an interface and allows two software applications to communicate with each other. API is a collection of software functions which can be executed by another software program.

## **What are the core components of an HTTP request?**

An HTTP request includes five key elements:

* HTTP methods – Set of request methods to perform desired action for a given resource (GET, PUT, POST, DELETE)
* Uniform Resource Identifier (URI) – Describes the resource
* HTTP Version, (example- HTTP v1.1)
* Request Headers, (example- Content-type : application/json, Content-Length : 511)
* Payload – It is basically a Request Body which includes message content.

## **State The Core Components of an HTTP Response?**

Every HTTP response contains four key elements.

* Status/Response Code – These are response codes issued by a server to a client’s request. For example, 404 means Page Not Found, and 200 means Response is OK.
* HTTP Version – describes HTTP version, for example-HTTP v1.1.
* Response Header – Includes information for the HTTP response message. For example, Content-type, Content-length, date, status and server type.
* Response Body – It contains the data that was requested by a client to server.

## **What is GUID?**

GUID solves the purpose of uniqueness. GUID stands for Global Unique Identifier. It is basically hexadecimal digits separated by hyphens.