Node Js basics

1. What are modules in Nodejs. Do you know any commonly used modules.

ANS - We use modules to break down large programs into small manageable and organized files. Furthermore, modules provide reusability of code. We can define our most used functions in a module and import it, instead of copying their definitions into different programs.
modules is a set of functions. Local modules are modules created locally in your Node. js application. These modules include different functionalities of your application in separate files and folders. You can also package it and distribute it via NPM, so that Node. Third-party modules are modules that are available online using the Node Package Manager(NPM). These modules can be installed in the project folder or globally. Some of the popular third-party modules are mongoose, express, angular, and react.

2. How do you write functions and global variables in a module A and use them in modlue B

3. How do you import an external package. What code do you use for this?

ANS - Command "require" is used in Node JS for import external libraries. Mentioned below is an example of this. "var http=require ("http")" . This will load the library and the single exported object through the HTTP variable.

4. What is express?

ANS - Express. js is a framework of Node. js which means that most of the code is already written for programmers to work with. ... js makes it easier to manage web applications.It is a part of a javascript based technology called MEAN software stack AN MERN SOFTWERE STACK which stands for MongoDB, ExpressJS, AngularJS, and Node.
Express.js is used by Fox Sports, PayPal, Uber and IBM .

5. How do you create routes in an express application?

ANS - we have to require express in our file module, then we have to access the router functionality with the help of express.router and we have to export the route to use in different file or folder.

6. What are middlewares and why do you need them?

ANS - Middleware functions are functions that have access to the request object (req), the response object (res), and the next function (next) in the application's request-response cycle. ... Middleware functions can perform the following tasks: Execute any code. Make changes to the request and the response objects. These functions are used to modify req and res objects for tasks like parsing request bodies, adding response headers, etc.

7. What are route and application level middlewares.

ANS -Express is a routing and middleware web framework that has minimal functionality of its own: An Express application is essentially a series of middleware function calls. ... Middleware functions can perform the following tasks: Execute any code. Make changes to the request and the response objects. Application level middleware are bound to an instance of express, using app. use() and app. VERB(). Router level middleware work just like application level middleware except they are bound to an instance of express.

8. How do you decide which code should go in an application level middleware vs which one is suitable for a route level middleware.

ANS - Types of express middleware which we have used. Application level middleware app.use Router level middleware router.use Built-in middleware express.static,express.json,express.urlencoded Error handling middleware app.use(err,req,res,next) Thirdparty middleware bodyparser

9. Name the middlewares you have used in your assignments and project implementation.

ANS -Application Level Middleware Example:
Authentication middleware Suppose we are having five routes getUsers,getDetails,updateDetails,isLoggedIn,isLoggedOut every route must be authenticated if the user is not authenticated then he is not able to call the above mentioned routes,so every GET,POST calls required authentication.In this case we build a authentication middleware. Now once the request comes the auth middleware will do some authentication logic that we have written inside it.Once authentication successful then remaining routed must be called using next() if auth fails then it wont perform next route exit the middleware with error response logic.

10. What is the main purpose of debugger in VS Code (or other IDEs).

ANS - A debugging tool can take you to the exact place in your code where the exception occurred and can help you investigate possible fixes.

11. How do you use a debugger. Show it for the current project you are working on.

ANS - Set a breakpoint and start the debugger To debug, you need to start your app with the debugger attached to the app process. ... Press F5 (Debug > Start Debugging) or the Start Debugging button Start Debugging in the Debug Toolbar, and the debugger runs to the first breakpoint that it encounters. If the app is not yet running, F5 starts the debugger and stops at the first breakpoint. To start your app with the debugger attached, press F11 (Debug > Step Into). F11 is the Step Into command and advances the app execution one statement at a time. Run to Click button is similar to setting a temporary breakpoint.

12. What all options are there in a debugger?

ANS - there are 6 options for debugger controller: 1. pause 2. go to next function 3. go to next line. 4. go to upper line 5. refresh or restart a debugger. 6. disconnect a debugger in our file or module.

13. What does synchronous mean?

ANS -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. Synchronous code is also called “blocking” because it halts the program until all the resources are available.

14. Where would you use synchronous functions?

ANS -javascript is a synchronous. as it always be in a single threaded.

15. What does asynchronous mean?

ANS - Asynchronous programming is a design pattern which ensures the non-blocking code execution. ... Asynchronous does exactly opposite, asynchronous code executes without having any dependency and no order. This improves the system efficiency and throughput.

16. Where would you use asynchronous functions?

ANS - Node. js uses callbacks, being an asynchronous platform, it does not wait around like database query, file I/O to complete. The callback function is called at the completion of a given task; this prevents any blocking, and allows other code to be run in the meantime.