Mern Backend Interview Questions /50

1. What are the key features of Node.js?

Answer : Let's look at some of the key features of Node.js.

- **Asynchronous event driven IO helps concurrent request handling –** All APIs of Node.js are asynchronous. This feature means that if a Node receives a request for some Input/Output operation, it will execute that operation in the background and continue with the processing of other requests. Thus it will not wait for the response from the previous requests.
- **Fast in Code execution –** Node.js uses the V8 JavaScript Runtime engine, the one which is used by Google Chrome. Node has a wrapper over the JavaScript engine which makes the runtime engine much faster and hence processing of requests within Node.js also become faster.
- **Single Threaded but Highly Scalable –** Node.js uses a single thread model for event looping. The response from these events may or may not reach the server immediately. However, this does not block other operations. Thus making Node.js highly scalable. Traditional servers create limited threads to handle requests while Node.js creates a single thread that provides service to much larger numbers of such requests.
- **Node.js library uses JavaScript –** This is another important aspect of Node.js from the developer's point of view. The majority of developers are already well-

versed in JavaScript. Hence, development in Node.js becomes easier for a developer who knows JavaScript.

- **There is an Active and vibrant community for the Node.js framework –** The active community always keeps the framework updated with the latest trends in the web development.
- **No Buffering –** Node.js applications never buffer any data. They simply output the data in chunks.

2. What do you mean by *Asynchronous API*?

Answer : All APIs of Node.js library are aynchronous that is non-blocking. It essentially means a Node.js based server never waits for a API to return data. Server moves to next API after calling it and a notification mechanism of Events of Node.js helps server to get response from the previous API call.

3. What is V8?

Answer : The V8 library provides Node.js with a JavaScript engine (a program that converts Javascript code into lower level or machine code that microprocessors can understand), which Node.js controls via the V8 C++ API. V8 is maintained by Google, for use in Chrome.

The Chrome V8 engine :

- The V8 engine is written in C++ and used in Chrome and Nodejs.
- It implements ECMAScript as specified in ECMA-262.
- The V8 engine can run standalone we can embed it with our own C++ program.

4. What is the file `package.json`??

Answer : All npm packages contain a file, usually in the project root, called `package.json` - this file holds various metadata relevant to the project. This file is used to give information to `npm` that allows it to identify the project as well as handle the project's dependencies. It can also contain other metadata such as a project description, the version of the project in a particular distribution, license

information, even configuration data - all of which can be vital to both `npm` and to the end users of the package. The `package.json` file is normally located at the root directory of a Node.js project.

Here is a minimal `package.json`:

```
{
  "name" : "barebones",
  "version" : "0.0.0",
}
```

5. Explain advantages of *BSON* over *JSON* in MongoDB?

Answer

- In addition to compactness, BSON adds additional data types unavailable in JSON, notably the BinData and Date data types.
- BSON is also designed to be fast to encode and decode. For example, integers are stored as 32 (or 64) bit integers, so they don't need to be parsed to and from text. This uses more space than JSON for small integers, but is much faster to parse.
- **BSON** is designed to be efficient in space, but in some cases is not much more efficient than JSON. In some cases BSON uses even more space than JSON. The reason for this is another of the BSON design goals: traversability. BSON adds some "extra" information to documents, like length of strings and subobjects. This makes traversal faster.

6. Name some *Built-in Globals* in Node.js?

Node.js has a number of built-in global identifiers that every Node.js developer should have some familiarity with. Some of these are true globals, being visible everywhere; others exist at the module level, but are inherent to every module, thus being pseudo-globals.

The list of **true globals**:

- `global` - The global namespace. Setting a property to this namespace makes it globally visible within the running process.

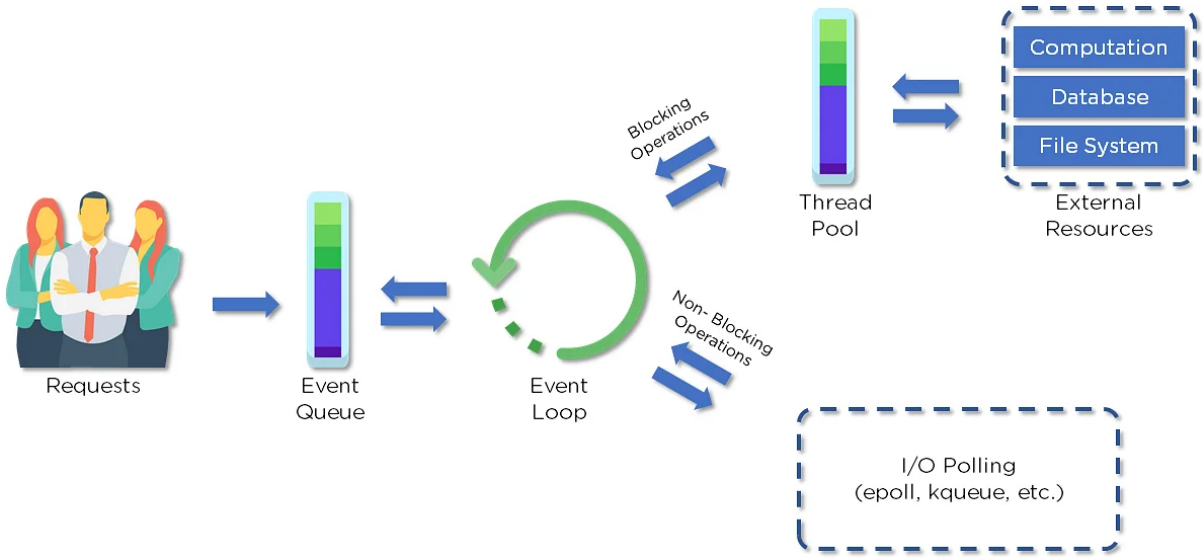
- `process` - The Node.js built-in `process` module, which provides interaction with the current Node.js process. Read More
- `console` - The Node.js built-in `console` module, which wraps various STDIO functionality in a browser-like way. Read More
- `setTimeout()`, `clearTimeout()`, `setInterval()`, `clearInterval()` - The built-in timer functions are globals. Read More

The **pseudo-globals** included at the module level in every module:

- `module`, `module.exports`, `exports` - These objects all pertain to the Node.js module system. Read More
- `__filename` - The `__filename` keyword contains the path of the currently executing file. Note that this is not defined while running the Node.js REPL.
- `__dirname` - Like `__filename`, the `__dirname` keyword contains the path to the root directory of the currently executing script. Also not present in the Node.js REPL.
- `require()` - The `require()` function is a built-in function, exposed per-module, that allows other valid modules to be included.

7. How does Node.js work?

A web server using Node.js typically has a workflow that is quite similar to the diagram illustrated below. Let's explore this flow of operations in detail.



- Clients send requests to the webserver to interact with the web application. Requests can be non-blocking or blocking:

- Querying for data
- Deleting data
- Updating the data
- Node.js retrieves the incoming requests and adds those to the Event Queue
- The requests are then passed one-by-one through the Event Loop. It checks if the requests are simple enough not to require any external resources
- The Event Loop processes simple requests (non-blocking operations), such as I/O Polling, and returns the responses to the corresponding clients

A single thread from the Thread Pool is assigned to a single complex request. This thread is responsible for completing a particular blocking request by accessing external resources, such as computation, database, file system, etc.

Once the task is carried out completely, the response is sent to the Event Loop that sends that response back to the client.

8. Why is Node.js Single-threaded?

Node.js is single-threaded for async processing. By doing async processing on a single-thread under typical web loads, more performance and scalability can be achieved instead of the typical thread-based implementation.

9. How is Node.js most frequently used?

Node.js is widely used in the following applications:

1. Real-time chats
2. Internet of Things
3. Complex SPAs (Single-Page Applications)
4. Real-time collaboration tools
5. Streaming applications
6. Microservices architecture

10. Explain the difference between frontend and backend development?

Front-end	Back-end
Frontend refers to the client-side of an application	Backend refers to the server-side of an application
It is the part of a web application that users can see and interact with	It constitutes everything that happens behind the scenes
It typically includes everything that attributes to the visual aspects of a web application	It generally includes a web server that communicates with a database to serve requests
HTML, CSS, JavaScript, AngularJS, and ReactJS are some of the essentials of frontend development	Java, PHP, Python, and Node.js are some of the backend development technologies

11. What is NPM?

NPM stands for Node Package Manager, responsible for managing all the packages and modules for Node.js.

Node Package Manager provides two main functionalities:

- Provides online repositories for node.js packages/modules, which are searchable on search.nodejs.org
- Provides command-line utility to install Node.js packages and also manages Node.js versions and dependencies

12. What does event-driven programming mean?

An event-driven programming approach uses events to trigger various functions. An event can be anything, such as typing a key or clicking a mouse button. A call-back function is already registered with the element executes whenever an event is triggered.

13. What is an Event Loop in Node.js?

Event loops handle asynchronous callbacks in Node.js. It is the foundation of the non-blocking input/output in Node.js, making it one of the most important environmental features.

14. Differentiate between process.nextTick() and setImmediate()?

The distinction between method and product. This is accomplished through the use of nextTick() and setImmediate(). next Tick() postpones the execution of action until the next pass around the event loop, or it simply calls the callback function once the event loop's current execution is complete, whereas setImmediate() executes a callback on the next cycle of the event loop and returns control to the event loop for any I/O operations.

15. For Node.js, why does Google use the V8 engine?

The V8 engine, developed by Google, is open-source and written in C++. Google Chrome makes use of this engine. V8, unlike the other engines, is also utilized for the popular Node.js runtime. V8 was initially intended to improve the speed of JavaScript execution within web browsers. Instead of employing an interpreter, V8 converts JavaScript code into more efficient machine code to increase performance. It turns JavaScript code into machine code during execution by utilizing a JIT (Just-In-Time) compiler, as do many current JavaScript engines such as SpiderMonkey or Rhino (Mozilla).

16. Explain the concept of middleware in Node.js.

Middleware is a function that receives the request and response objects. Most tasks that the middleware functions perform are:

- Execute any code
- Update or modify the request and the response objects
- Finish the request-response cycle
- Invoke the next middleware in the stack

17. What are the different types of HTTP requests?

HTTP defines a set of request methods used to perform desired actions. The request methods include:

GET: Used to retrieve the data

POST: Generally used to make a change in state or reactions on the server

HEAD: Similar to the GET method, but asks for the response without the response body

DELETE: Used to delete the predetermined resource

18. What is CORS? Why is it important?

Ans. This is an important ***web development interview question***.

Cross-Origin Resource Sharing (CORS) is a browser mechanism that allows controlled access to resources located outside of a given domain. It enables a web page from one domain to access a resource with a different domain (a cross-domain request). It is a relaxation of the same-origin policy implemented in modern browsers.

Due to the same-origin policy followed by XMLHttpRequest and fetch, JavaScript can only make calls to URLs that live on the same origin as the location where the script is running. Without features like CORS, websites are restricted to accessing resources from the same origin through the same-origin policy.

19. What are the advantages of HTTP 2.0 over HTTP 1.1?

Ans. The major advantages of HTTP 2.0 over HTTP 1.1 include **–**

- Higher loading speed
- Improvement of web positioning
- Automatic prioritization
- Less broadband consumption
- Immediate presentation

20. How do you take into account SEO, maintainability, UX, performance, and security when you’re building a web application?

Ans. Explain how you prioritize your actions as per the requirements of the organization. If your organization handles vital data, then security will be your top priority. If it is a medium-sized online business, SEO and UX might be your top priority, and so on.

21. What are the new form elements introduced in HTML5?

Ans. The new form elements introduced in HTML5 are:

- <datalist> – specifies a list of options for input controls
- <keygen> – generates an encryption key
- <output> – defines the result of an expression
- <progress> – heads only in the direction of 100% of the max value
- <meter> – provides for a gauge, displaying a general value within a range

22. What is your preferred development environment?

Ans. This question is not about checking if you are perfect for the same environment as the organization works but to measure if you are flexible to work in any environment. So, give them a hint that you are able to adapt to any environment with the core skills that you have.

23. Which are the new APIs provided by HTML5?

Ans. The new APIs are –

- Media API
- Text track API
- Application cache API
- Data transfer API
- User interaction API
- Command API
- Constraint validation API
- History API

24. By which mechanism in JavaScript can you detect the operating system on a client machine?

Ans. The operating system on a client system can be known by using the JavaScript property navigator.appVersion.

25. What is Type Coercion in JavaScript?

Ans. Type coercion refers to the conversion of a value from one type to another (e.g Number to String, String to Number, or Boolean to Number) with similar content. In case the behavior of the implicit conversion is not sure, then the constructors of a data type can be used to convert any value to that datatype.

26. What is Webpack?

Ans. Webpack is a static module bundler for JavaScript. It is a build tool that is used is to bundle JavaScript files for usage in a browser. It puts the bundles of assets, such as codes, images, fonts, and files in a dependency graph, and enables you to use require() in your source code to point to local files, like images, and decide how they’re processed in the final Javascript bundle.

While a webpack may slow you down at the beginning, it can give you great speed benefits when used correctly.

27. What is progressive rendering in HTML?

Ans. Progressive Rendering or Progressive Server Side Rendering is a technique with which you can sequentially update small parts of the entire webpage and stream it to the client in parts without waiting for the whole page to be rendered.

It means that when you start rendering the critical content on the server, you can stream it to the client without waiting for non-critical content to be rendered. It bridges the benefits of both CSR (Client Side Rendering) and SSR (Server Side Rendering).

28. Explain the functional and non-functional requirements?

Ans. Functional requirements define the specific functionality of the system, It describes what the system does or must not do.

Non-functional requirements, on the other hand, define how the system should do it. It specifies a system's type, in terms of accessibility, reliability, capacity, usability, maintainability, and security. Non-functional requirements describe system behavior, features, and general characteristics that affect the user experience.

Non-functional requirements do not affect the basic functionality of the system. The system will continue to perform its basic purpose, even if the non-functional requirements are not met.

29. What is long polling?

Ans. Long polling is a technique of having a persistent connection with the server. In long polling, the client polls the server requesting new information. The server holds

a client’s connection open for as long as possible. The connection is closed only after the data is sent back to the client or connection timeout occurs.

Long Polling Flow:

- A request is sent to the server.
- The server holds a client’s connection until new data is available.
- The server responds to the request when the new information appears.
- The browser immediately sends another request.

30. Explain what is the difference between local storage and cookies?

Ans. The differences between local storage and cookies are:

Local Storage	Cookies
Local Storage is for the client-side.	These are for the client as well as the server-side.
It is larger and can hold information on the client-side.	They are smaller and send data to the server-side with every HTTP request
Storage capacity is 5MB per domain.	Storage capacity is 4095 bytes/cookie.
It does not have an expiration and has to be removed manually.	They have an expiration. Cookie data get deleted after some time. You can set the expiration duration.

31. How to handle type conversion in JavaScript?

Ans. JavaScript is a weakly typed language. It means that whenever an operator or statement expects a particular data-type, JavaScript automatically converts the data to that type. Javascript supports automatic type conversion. It is the common way of conversion of types used by JavaScript developers.

32. How can you reduce page loading time?

Ans. These are the following ways you can reduce web page loading time –

- Reduce the image size
- Use the latest generation formats for images
- Minify HTML, CSS, and Javascript
- Postpone uploading off-screen images
- Create Accelerated Mobile Pages (AMPs)
- Remove unnecessary widgets
- Avoid multiple redirects
- Place CSS at the top and script referencing at the bottom or external files
- Reduce lookups
- Minimize redirects and caching
- Check the current speed of the website
- Finding a good hosting to host your website
- Clean the web code

33. Name the different formats for data exchange.

Ans. Different formats used for data exchange include XML, JSON, CSV, and Text formats.

34. What is Ajax?

Ans. AJAX stands for Asynchronous JavaScript and XML. It enables applications to transport data to or from a server asynchronously without refreshing the page. It means that parts of a web page are updated, without reloading the entire page. AJAX can be used anywhere in a web application where small amounts of information could be saved or retrieved from the server without posting back the entire page. This technique helps in creating better, faster, and interactive web applications. Below are some benefits of AJAX:

- Helps in performing perform a callbacks
- Allows us to make asynchronous calls to a web server

- Improved the speed and usability of a web application
- User-friendly

35. What is Event Bubbling?

Ans.

Event Bubbling is a type of event propagation in which one element is nested inside a second element, and both elements have registered a handle to that event. This process starts with the element that triggers the event and then bubbles up to the containing elements in the hierarchy. The event is first captured and handled by the innermost element and then successfully propagated to outer elements. Thus, when an event happens on an element, it first runs the handlers on it, then on its parent, then all other ancestors.

36. Is node js entirely single-threaded?

Yes, node js is single-threaded but some behavior of node js makes our code fast which is asynchronous behavior

37. What is control flow function?

A block of code or piece of code that runs in between an async function call is called a control flow function

38. What types of tasks we can handle asynchronously?

1. I/o Operations .
2. Heavy task
3. Blocking task.

39. What you used for authentication purpose?

JSON web token

40. What is express js?

1. Express js is a framework of nodejs.
2. Non-blocking servers that can handle requests better.
3. Handle req/res cycle.
4. Rapid server side programming
5. Express acts as middleware.

41. Express features?

1. Follow mvc architecture
2. Error handling middleware.

42. Types of middleware in express ?

1. Application level middleware. E.g auth middleware
2. Router level middleware .
3. Error handling middleware
4. Third party middleware e.g body-parser, cookie.

43. What is helmet middleware in express js ?

Helmet is an express middleware, it has various HTTP response headers. The main purpose of helmet middleware is to secure the get and post requests in the node js application.

44. What is the purpose of app.use()?

App. use() we register the middleware in our app app. use(middlware) is called every time a request is sent to the server. Request: e.g get, post, delete, put.

45. What is payload in rest api.?

Request data that is present in the body part.

46. What is postman api tool.?

It is a tool to develop and test API workflow.

47. What is Promise, async, await, then.?

Promise:

Promise is an object that encapsulates our result in an asynchronous function.

Async :

Async is a function that handles asynchronous operations.

Await :

We used to wait for promises.

Then:

We used to get the value of the result.

48. What is event loop?

The event loop makes the code async. Behind the seen when our code run asynchronously then there are the containers that are call-stack, call-back queue, web API, and micro-task queue. The purpose of the event loop is to execute the async task as per priority. E.g I have 5 statements of code and statement 1 and 5 is taking time so statement 2, 3, and 4 will push on the call stack, and 1, and 5 statements will go into the web API. Then 1, and 5 will go into the callback queue. Whenever the main call stack will pop all 3 statements then the event loop will start and allow a callback queue

to push statements 1 and 5 in the main call stack and by this, our program will be asynchronous and there will be no blocking in the program.

49. When to use node js and when not?

We can build an application when our task is like a chatbot, streaming, IoT based. And when we need dedicate time to the CPU then we shouldn’t use nodejs as you know nodejs is single-threaded.

50. What do you use to start script in development and production deployment?

In the development process, we used the nodemon package and start the script “dev”: “nodemon main-file.js” run => npm run dev. And in the production deployment process, we used “start”: “node main-file.js”.