Most common MERN Stack Interview Questions with Answers

In this Document we have Covered:

1- Mongo DB Interview Questions And Answers

1- What are Indexes in MongoDB?

Indexes support the efficient execution of queries in MongoDB. Without indexes, MongoDB must perform a collection scan, i.e. scan every document in a collection, to select those documents that match the query statement. If an appropriate index exists for a query, MongoDB can use the index to limit the number of documents it must inspect.
Indexes are special data structures that store a small portion of the collection's data set in an easy-to-traverse form. The index stores the value of a specific field or set of fields, ordered by the value of the field. The ordering of the index entries supports efficient equality matches and range-based query operations. In addition, MongoDB can return sorted results by using the ordering in the index.

2- How many indexes does MongoDB create by default for a new collection?

By default, MongoDB creates a unique index on the _id field during the creation of a collection. The _id index prevents clients from inserting two documents with the same value for the _id field.

3- What is "Namespace" in MongoDB?

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

4- What is Replication in MongoDB?

Replication exists primarily to offer data redundancy and high availability. It maintain the durability of data by keeping multiple copies or replicas of that data on physically isolated servers. Replication allows to increase data availability by creating multiple copies of data across servers. This is especially useful if a server crashes or hardware failure.
With MongoDB, replication is achieved through a Replica Set. Writer operations are sent to the primary server (node), which applies the operations across secondary servers, replicating the data. If the primary server fails (through a crash or system failure), one of the secondary servers takes over and becomes the new primary node via election. If that server comes back online, it becomes a secondary once it fully recovers, aiding the new primary node.

5- How can you achieve primary key - foreign key relationships in MongoDB?

The primary key-foreign key relationship can be achieved by embedding one document inside another. As an example, a department document can have its employee document(s).

6- When should we embed one document within another in MongoDB?

You should consider embedding documents for:

- *contains* relationships between entities
- One-to-many relationships
- Performance reasons

7- Explain the limitations of MongoDB Transactions?

MongoDB transactions can exist only for relatively short time periods. By default, a transaction must span no more than one minute of clock time. This limitation results from the underlying MongoDB implementation. MongoDB uses MVCC, but unlike databases such as Oracle, the “older” versions of data are kept only in memory.

- You cannot create or drop a collection inside a transaction.
- Transactions cannot make writes to a capped collection
- Transactions take plenty of time to execute and somehow they can slow the performance of the database.
- Transaction size is limited to 16MB requiring one to split any that tends to exceed this size into smaller transactions.

Gaurav Pandey

Swipe>>

- Subjecting a large number of documents to a transaction may exert excessive pressure on the WiredTiger engine and since it relies on the snapshot capability, there will be a retention of large unflushed operations in memory. This renders some performance costs on the database.

8- Is there an "upsert" option in the MongoDB insert command?

The db.collection.insert() provides no upsert possibility. Instead, mongo insert inserts a new document into a collection. Upsert is only possible using db.collection.update() and db.collection.save().

2- Express Interview Questions And Answers

1- What is Scaffolding in Express.js?

Scaffolding is creating the skeleton structure of the application
There are 2 ways to do this, by using:

1. Express Application Generator
2. Yeoman

2- How to enable debugging in the express app?

In different Operating Systems, we have the following commands:

On Linux:

```
DEBUG=express:*
node app.js
```

On Windows:

```
set DEBUG=express:*
node app.js
```

Gaurav Pandey

Swipe>>

3- Serving static files in Express.js?

```
app.use(express.static('public'))
app.use('/static', express.static(path.join(__dirname, 'public')));
```

4- Database integration in express.js?

Express.js supports many RDBMS & NoSQL databases like

- MongoDB
- MySQL
- Oracle
- PostgreSQL
- SQL Server
- SQLite

#Example: Install MongoDB

```
>>npm install MongoDB

var MongoClient = require('mongodb').MongoClient

MongoClient.connect('mongodb://localhost:27017/test_db', function (err, db) {
  if (err) throw err

  db.collection('mammals').find().toArray(function (err, result) {
    if (err) throw err

    console.log(result)
  })
})
```

Gaurav Pandey

Swipe>>

5- Error handling in Express.js?

```
var express = require('express'),
app = express();

app.use(function (err, req, res, next) {
  console.error(err.stack) // error first callback
  res.status(500).send('Something went wrong!')
})
```

6- How dynamic routing works in express.js?

When someone passes parameters in URL (i.e. Parametrized URL), this routing phenomenon is called dynamic routing.

```
var express = require('express'),
app = express();

app.get('/article/:id', function(req , res){
  res.render('article' + req.params.id);
})
```

In the above example: id is a parameter, which can be different for different requests.

7- What is routing and how routing works in Express.js?

Routing refers to determining how an application responds to a request.

Route Syntax:

```
app.METHOD(PATH, HANDLER);
```

Where:

- **the app** is an instance of **express**.
- **METHOD** is an HTTP request method (get, post, put, etc.).
- **PATH** is a path/endpoint on the server.
- **HANDLER** is a function executed when the route is matched.

#Example: A route with path / and get method.

```
app.get('/', function (req, res) {
  res.send('Express.js Interview Questions')
})
```

3- React js Interview Questions And Answers

1- What is React?

React is an open-source front-end JavaScript library that is used for building user interfaces, especially for single-page applications. It is used for handling the view layer for web and mobile apps. React was created by Jordan Walke, a software engineer working for Facebook. React was first deployed on Facebook's News Feed in 2011 and on Instagram in 2012.

2- What are the major features of React?

The major features of React are:

- It uses VirtualDOM instead of RealDOM considering that RealDOM manipulations are expensive.
- Supports server-side rendering.
- Follows Unidirectional data flow or data binding.
- Uses reusable/composable UI components to develop the view.

3. When to use a Class Component over a Function Component?

If the component needs *state or lifecycle methods* then use class component otherwise use function component. *However, from React 16.8 with the addition of Hooks, you could use state , lifecycle methods, and other features that were*

only available in class component right in your function component. *So, it is always recommended to use Function components, unless you need a React functionality whose Function component equivalent is not present yet, like Error Boundaries *

4- What are Pure Components?

React.PureComponent is exactly the same as *React.Component* except that it handles the shouldComponentUpdate() method for you. When props or state changes, *PureComponent* will do a shallow comparison on both props and state. *Component* on the other hand won't compare current props and state to next out of the box. Thus, the component will re-render by default whenever shouldComponentUpdate is called.

5- What are props in React?

Props are inputs to components. They are single values or objects containing a set of values that are passed to components on creation using a naming convention similar to HTML-tag attributes. They are data passed down from a parent component to a child component.

The primary purpose of props in React is to provide following component functionality:

i. Pass custom data to your component.
ii. Trigger state changes.
iii. Use via this.props.reactProp inside component's render() method.

For example, let us create an element with reactProp property:

```
<Element reactProp={'1'} />
```

This reactProp (or whatever you came up with) name then becomes a property attached to React's native props object which originally already exists on all components created using React library.

```
props.reactProp
```

6- What is the difference between state and props?

Both *props* and *state* are plain JavaScript objects. While both of them hold information that influences the output of render, they are different in their functionality with respect to components. Props get passed to the component similar to function parameters whereas state is managed within the component similar to variables declared within a function.

7- What is the purpose of the callback function as an argument of setState()?

The callback function is invoked when setState is finished and the component gets rendered. Since setState() is asynchronous the callback function is used for any post action.

Note: It is recommended to use the lifecycle method rather than this callback function.

setState({ name: 'John' }, () => console.log('The name has updated and component re-rendered'))

8- What are synthetic events in React?

SyntheticEvent is a cross-browser wrapper around the browser's native event. Its API is the same as the browser's native event, including stopPropagation() and preventDefault(), except the events work identically across all browsers.

9- What is the use of refs?

The *ref* is used to return a reference to the element. They *should be avoided* in most cases, however, they can be useful when you need direct access to the DOM element or an instance of a component.

10- What is Virtual DOM?

The ***Virtual DOM*** **(VDOM)** is an in-memory representation of the *Real DOM*. The representation of a UI is kept in memory and synced with the "real" DOM. It's a step that happens between the render function being called and the displaying of elements on the screen. This entire process is called *reconciliation*.

11- What is the difference between Shadow DOM and Virtual DOM?

The *Shadow DOM* is a browser technology designed primarily for scoping variables and CSS in *web components*. The *Virtual DOM* is a concept implemented by libraries in JavaScript on top of browser APIs.

12- What is React Fiber?

Fiber is the new *reconciliation* engine or reimplementation of the core algorithm in React v16. The goal of React Fiber is to increase its suitability for areas like animation, layout, gestures, ability to pause, abort, or reuse work and assign priority to different types of updates; and new concurrency primitives.

4- Node js Interview Questions And Answers

1- What is Node.js? Where can you use it?

Node.js is server-side scripting based on Google's V8 JavaScript engine. It is used to build scalable programs, especially web applications that are computationally simple but are frequently accessed.
You can use Node.js in developing I/O intensive web applications like video streaming sites. You can also use it for developing: Real-time web applications, Network applications, General-purpose applications, and Distributed systems.

2- Which types of applications were developed using Node js?

You can develop Real-time web applications, Network applications, Distributed Systems, and General-purpose applications using Node js.

3- What is the difference between Node.js vs Ajax?

The difference between Node.js and Ajax is that Ajax (short for Asynchronous Javascript and XML) is a client-side technology, often used for updating the contents of the page without refreshing it. While,Node.js is Server Side Javascript, used for developing server software. Node.js does not execute in the browser but by the server.

4- What are the functionalities of NPM in Node.js?

NPM (Node Package Manager) provides two functionalities:

An online repository for Node.js packages
Command-line utility for installing packages, version management and dependency management of Node.js packages

5- In which Language Node Js is written?

Node js is written in C, C++,JavaScript.It uses Google's open-source V8 Javascript Engine to convert Javascript code to C++. (node js interview questions and answers pdf)

6- Why should you use Node.js?

Scalable network programs can be developed easily by Node.js and if you like to know why we should use Node, js then due to the below-listed advantages it is usually used by the organizations:
- Great concurrency is yielded by Node.js
- Every feature of Node.js is asynchronous
- It is never blocked
- It is quite faster
- A unified programming language and data type is offered by this

7- Explain various streams of Node.js?

In Node.js stream allow users to read data from the source and o write data to a destination in a continuous process.
They are just an object and the following four types of streams are there in **Node.js they are:**

- To provide read operation
- To provide a write operation
- To provide both read and write operation
- A duplex stream form that can perform computations as per available data.

8- 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 which are searchable on search.nodejs.org
Command-line utility to install packages, do version management and dependency management of Node.js packages.

9- Which Is The First Argument Usually Passed To A Node.Js Callback Handler?

Node.js core modules follow a standard signature for their callback handlers and usually the first argument is an optional error object. And if there is no error, then the argument defaults to null or undefined.

10- What is the difference between AngularJS and Node.js?

Angular.JS is a web application development framework while Node.js is a runtime system.

Reference:

https://github.com/learning-zone/mongodb-interview-questions
https://www.fullstacktutorials.com/interviews/top-10-express-js-interview-questions-answers-30.html
https://github.com/sudheerj/reactjs-interview-questions
https://github.com/learning-zone/nodejs-interview-questions

Gaurav Pandey

Swipe>>