**-------------------------------EXPRESS-------------------------------**

1. ***How does express work?***

* Express is a nodejs web application framework that provides broad features for building web and mobile applications. It is used to build a single page, multipage, and hybrid web application.
* It's a layer built on the top of the Node js that helps manage servers and routes.

1. ***What are routes?***

* It refers to how an application's endpoint's URLs respond to client requests.

1. ***What are Middlewares?***

* Middleware is a request handler that has access to the application's request-response cycle.

1. ***What is MVC framework?***

* MVC stands for Model, View, Controller is an architectural pattern that separates an application into three main logical components: the model, the view, and the controller. Each one of these components is built to handle specific development aspects of an application.
* *Controller* is the part that takes care of client request processing which handles the HTTP Request and returns a response
* *Model* is the database interface which lets you interact with the database API and create different entity schemas of your app on the database.
* *View* is what compiles and renders into plain HTML and what the client in most cases going to get back as a response of what he requested.

1. ***How do you do validations?***
2. ***How do you do static routing?***

* Static Routing is also known as non-adaptive routing which doesn’t change the routing table unless the network administrator changes or modifies them manually.

|  |  |
| --- | --- |
| **Static Routing** | **Dynamic Routing** |
| In static routing routes are user-defined. | In dynamic routing, routes are updated according to the topology. |
| Static routing does not use complex routing algorithms. | Dynamic routing uses complex routing algorithms. |
| Static routing provides high or more security. | Dynamic routing provides less security. |
| Static routing is manual. | Dynamic routing is automated. |
| Static routing is implemented in small networks. | Dynamic routing is implemented in large networks. |
| In static routing, additional resources are not required. | In dynamic routing, additional resources are required. |
| In static routing, failure of the link disrupts the rerouting. | In dynamic routing, failure of the link does not interrupt the rerouting. |
| Less Bandwidth is required in Static Routing. | More Bandwidth is required in Dynamic Routing. |
| Static Routing is difficult to configure. | Dynamic Routing is easy to configure. |
| Another name for static routing is non-adaptive routing. | Another name for dynamic routing is adaptive routing. |

1. ***What are some templating engines?***

* Template engines are used when you want to rapidly build web applications that are split into different components. Templates also enable fast rendering of the server-side data that needs to be passed to the application.
* The popular ones include Ejs, Jade, Pug, Mustache, HandlebarsJS, Jinja2, and Blade.
* When you build a server-side application with a template engine, the template engine replaces the variables in a template file with actual values, and displays this value to the client.

1. ***How do you manage sessions in express?***

* When the client makes a login request to the server, the server will create a session and store it on the server-side. When the server responds to the client, it sends a cookie. This cookie will contain the session’s unique id stored on the server, which will now be stored on the client. This cookie will be sent on every request to the server.
* We use this session ID and look up the session saved in the database or the session store to maintain a one-to-one match between a session and a cookie. This will make HTTP protocol connections stateful.

1. ***How do you manage cookies with express?***

* Cookies are small piece of information i.e. sent from a website and stored in user's web browser when user browses that website. Every time the user loads that website back, the browser sends that stored data back to website or server, to recognize user.
* You have to acquire cookie abilities in Express.js. So, install cookie-parser middleware through npm.

1. ***What are common libraries you work with express?***

* Cors- node.js package for providing a Connect/Express middleware that can be used to enable CORS with various options
* Axios-Promise based HTTP client for the browser and node.js
* Multer-For handling multipart/form-data, which is primarily used for uploading files.
* Morgan-HTTP request logger middleware for node.js
* Dotenv-For using environment variables from .env file
* Nodemailer-For sending e-mails
* Passport- Express-compatible authentication middleware for Node.js.
* Mongoose
* jest-For testing

1. ***What is CORS?***

* Cross-origin resource sharing (CORS) is a browser mechanism which enables controlled access to resources located outside of a given domain.

**----------------------------MONGOOSE----------------------------**

1. ***What are Models?***

* A Mongoose schema defines the structure of the document, default values, validators, etc., whereas a Mongoose model provides an interface to the database for creating, querying, updating, deleting records, etc.

1. ***Explain why mongoose does not return a promise but has a .then***

* Mongoose async operations, like .save() and queries, return thenables. This means that you can do things like MyModel.findOne({}).then() and await MyModel.findOne({}).exec() if you're using async/await.
* Mongoose queries are not promises. They have a .then() function for co and async/await as a convenience. If you need a fully-fledged promise, use the .exec() function.

1. ***What are aggregation pipelines with mongoose?***

* An aggregation pipeline consists of one or more stages that process documents:
* Each stage performs an operation on the input documents. For example, a stage can filter documents, group documents, and calculate values.
* The documents that are output from a stage are passed to the next stage.
* An aggregation pipeline can return results for groups of documents. For example, return the total, average, maximum, and minimum values.

1. ***I'm using an arrow function for a virtual, middleware, getter/setter, or method and the value of this is wrong. Why?***

* Arrow functions don't have their own bindings to this, arguments, or super, and should not be used as methods.
* Arrow functions cannot be used as constructors. Calling them with new throws a TypeError. They also don't have access to the new.target keyword.
* Arrow functions cannot use yield within their body and cannot be created as generator functions.

1. ***Should I create/destroy a new connection for each database operation?***
2. ***My query/update seems to execute twice. Why is this happening?***
3. ***How do you create indexes with mongoose***
4. ***What are pre and post hooks?***