What do you mean by Single Page Application?

A single page application is a web app or site which only loads a page once, and then dynamically “updates” the page. Any interactions with the page or subsequent pages do not require a round trip to the server, which means the page is not reloaded. This provides a user experience that is similar to a desktop or native app.

A SPA works by performing the user interface logic in the web browser by communicating with application programming interfaces (API). To put it simply, a single page application is an app that works inside a web browser, and it doesn’t require the page to reload when a user is using it.
Single page applications are a great way to create engaging and exceptional experiences for your website visitors. This is why many of the most engaging apps and webpages we use are likely to be a single page application.

2.What is MEAN stack?

MEAN Stack is one of the most popular Technology Stack. It is used to develop a Full Stack Web Application. Although it is a Stack of different technologies, all of these are based on JavaScript language.

MEAN Stands for:

1. **M** – MongoDB
2. **E** – Express
3. **A** – Angular
4. **N** – Node.js

This stack leads to faster development as well as the deployment of the Web Application. Angular is Frontend Development Framework whereas Node.js, Express, and MongoDB are used for Backend development as shown in the below figure.

Flow of Data in MEAN Stack Application: Here, each module communicates with the others in order to have a flow of the data from Server/Backend to Client/Frontend.

3.What is the role of Node.js in MEAN stack? || What are the key features of Node.js?

Node.js provides a JavaScript Environment which allows the user to run their code on the server (outside the browser). Node pack manager i.e. npm allows the user to choose from thousands of free packages (node modules) to download.

Why use Node.JS/feature of node js:

- Open-source JavaScript Runtime Environment
- Single threading – Follows a single-threaded model.
- Data Streaming
- Fast – Built on Google Chrome’s JavaScript Engine, Node.js has a fast code execution.
- Highly Scalable
- Initialize a Node.js application by typing running the below command in the command window. Accept the standard settings.

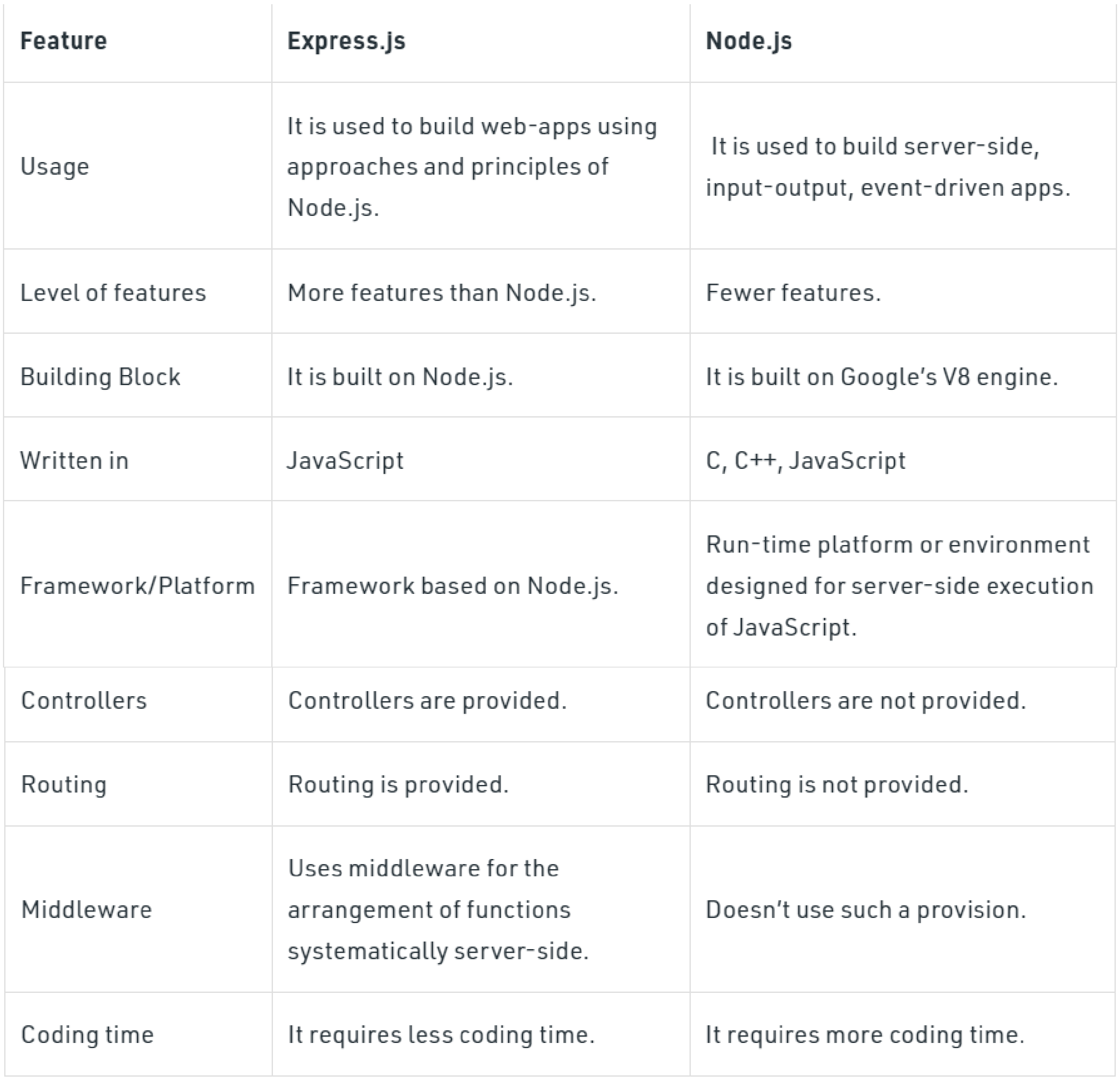
What do you understand by scaffolding in Express.js?

Scaffolding is creating the skeleton structure of application. It allows users to create own public directories, routes, views etc. Once the structure for app is built, user can start building it. Express is the open source web development framework for Node.js for building web applications and the APIs. To install **express** in the Node.js environment use the NPM (Node Package Manager).

Syntax:

npm install express --save

node js vs express js



Feature	Express.js	Node.js
Usage	It is used to build web-apps using approaches and principles of Node.js.	It is used to build server-side, input-output, event-driven apps.
Level of features	More features than Node.js.	Fewer features.
Building Block	It is built on Node.js.	It is built on Google's V8 engine.
Written in	JavaScript	C, C++, JavaScript
Framework/Platform	Framework based on Node.js.	Run-time platform or environment designed for server-side execution of JavaScript.
Controllers	Controllers are provided.	Controllers are not provided.
Routing	Routing is provided.	Routing is not provided.
Middleware	Uses middleware for the arrangement of functions systematically server-side.	Doesn't use such a provision.
Coding time	It requires less coding time.	It requires more coding time.

What is node.js and npm?

sometimes, Node.js is also called simply **Node** or **node.**

Nodejs: Node.js is a framework that contains the **V8** JavaScript engine, the standard library of packages, and some binaries. In reality, it is more complex than that as explained in the below diagram (*follow the link for more details*). Node.js uses different threads to perform low-level non-JavaScript time-taking operations. This way, our JavaScript is not blocked while a time taking operation like **reading a file** is in progress. Since these operations are running in the background once initiated, we need a confirmation or a **callback** when the operation is finished. This callback is a JavaScript function that will execute as soon as the operation is finished.

NPM: NPM – or "Node Package Manager" – is the default package manager for JavaScript's runtime Node.js.
NPM consists of two main parts:

- a CLI (command-line interface) tool for publishing and downloading packages, and
- an online repository that hosts JavaScript packages

package.json will be generated when npm init is run to initialise a JavaScript/Node.js project, with these basic metadata provided by developers:

- name: the name of your JavaScript library/project
- version: the version of your project. Often times, for application development, this field is often neglected as there's no apparent need for versioning opensource libraies. But still, it can come handy as a source of the deployment's version.
- description: the project's description
- license: the project's license

what do you mean by JavaScript closures ?

JavaScript closures Closures are functions that refer to variables from their parent environment. Using the closure pattern enables variables from the parent() function to remain bound to the closure. Let's take a look at the following example: function parent() { var message = "Hello World"; function child() { alert (message); } child(); } parent();

In the preceding example, you can see how the child() function has access to a variable deined in the parent() function. But this is a simple example, so let's see a more interesting one: function parent() { var message = 'Hello World'; function child() { alert (message); } return child; } var childFN = parent() childFN();

Closures are very important in asynchronous programming because JavaScript functions are irst-class objects that can be passed as arguments to other functions. This means that you can create a callback function and pass it as an argument to an event handler. When the event will be emitted, the function will be invoked, and it will be able to manipulate any variable that existed when the callback function was created even if its parent function was already executed. This means that using the closure pattern will help you utilize event-driven programming without the need to pass the scope state to the event handler.

How express handles the requests in MEAN project

Express.js is the most powerful framework of the node.js. Express.js is a routing and Middleware framework for handling the different routing of the webpage, and it works between the request and response cycle. Express.js use different kinds of middleware functions in order to complete the different requests made by the client for e.g. client can make get, put, post, and delete requests these requests can easily handle by these middleware functions. **andling Multiple requests using Express.js:**

Express.js contains multiple methods to handle all types of requests rather than work on a single type of request as shown below:

- **Express.js req.get() Method:** This method is used when get request is done by the client for e.g Redirecting another webpage requests etc
- **Express.js req.post() Method:** This method is used when post requests are done by the client for e.g. uploading documents etc.

- **Express.js req.delete() Method:** This method is used when a delete request is done by the client it is mainly done by the admin end for e.g. deleting the records from the server.
- **Express.js req.put() Method:** This method is used when update requests are done by the client to update the information over the website.

What is MongoDB and MongoDB shell?

MongoDB is an open-source document-oriented database that is designed to store a large scale of data and also allows you to work with that data very efficiently. It is categorized under the NoSQL (Not only SQL) database because the storage and retrieval of data in the MongoDB are not in the form of tables.

The MongoDB database is developed and managed by MongoDB.Inc under SSPL(Server Side Public License) and initially released in February 2009. It also provides official driver support for all the popular languages like C, C++, C#, and .Net, Go, Java, Node.js, Perl, PHP, Python, Motor, Ruby, Scala, Swift, Mongoid. So, that you can create an application using any of these languages. Nowadays there are so many companies that used MongoDB like Facebook, Nokia, eBay, Adobe, Google, etc. to store their large amount of data.

MongoDB Shell is the quickest way to connect, configure, query, and work with your MongoDB database. It acts as a command-line client of the MongoDB server.

The MongoDB Shell is a standalone, open-source product and developed separately from the MongoDB Server under the Apache 2 license. It is a fully functional JavaScript and Node.js 14.x REPL for interacting with MongoDB servers.

MongoDB Shell is already installed with MongoDB. You can find it in the installation directory where you installed MongoDB. By default, it is "C:\Program Files\MongoDB\Server". Open the installation folder and appropriate version folder and go to the "bin" folder. Here, "mongo.exe" is MongoDB shell. Click on it to open the MongoDB shell, as shown below.

Why is Node.js preferred over other backend technologies like Java and PHP

1.Single Page Applications are those that load a single HTML page and update it dynamically as user interaction with the app happens. This reduces response time and SPAs content appears easy for the users to interact with the application. Since NodeJS allows server-side rendering using which page can be rendered before it hits the browser. A few benefits of SPAs are SEO Optimization, fast and flexible, less loading time, and ease of navigation.

2. Data Streaming:Companies like Netflix use NodeJS for streaming data because of its lightweight and fast processing feature. The streams allow users to pipe requests to each other, which results in streaming data to its endpoint. NodeJS handles data streams with I/O bound apps. The process follows file uploading and data coming in a stream and then we process it online. This could also be done for real-time audio or video streaming. There are four types of streams -writable (write data), readable (read data), duplex (write and read data), and transform (convert read to write data, and vice versa).

3. Real-Time Apps:Real-time apps are those that allow two-way communication between clients and servers. Using NodeJS with Web-Sockets helps developers easily create one. NodeJS eases handling multiple requests made by the client and enables code sharing and re-usage of library codes. The apps get immediate responses and work within the limited time frame. Its single-threaded functionality makes it best suited for real-time communication. It is the perfect platform to create low-latency apps so it’s best when we have to process a high volume of short messages and uses socket.io which makes the creation of straightforward apps.

Nodejs vs Java: testing

With its extensive ecosystem of third-party packages, Nodejs provides competent testing and troubleshooting abilities. Various test automation tools/frameworks for Nodejs apps exist, including Mocha, Jest, Lab and Code, Jasmine, and AVA. Developers can use Java to create test scenarios. As a result, your teammates will be able to create versatile tests that include sequencing, grouping, and data-driven features. It also makes writing parallel tests easier. JUnit, Selenium, TestNG, Apache JMeter, are among the testing frameworks and tools supported.

. Nodejs vs Java: scalability

Microservices and Nodejs both create smaller portions of services and code modules. Because it is excellent at managing multiple simultaneous requests, Nodejs is suitable for constructing scalable apps. As a result, it’s the ideal combination for creating enterprise-grade sophisticated apps with increased scalability. Because of annotation syntax and Java frameworks like Spring Boot, Java is suitable with microservices.

Conclusion: Coding in PHP can be a good thing when you need a quick solution. But if you need a more efficient, complex, and sustainable output, you should opt for Node.js and invest more time in writing and compiling your code.

1. **Performance** : php Waits until the file system opens and reads the requested file.
1. **But nodejs :** Works on the next request without waiting for the previous one.