ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 4, April 2018

Analysis of Stack Technology: a Case study of MEAN vs. MERN Stack

Pragati Bhardwaj

Department of Computer Science & Engineering, Shri Ram College of Engineering & Management, Palwal, Haryana, India

ABSTRACT*:* For large enterprise web apps, taking advantage of web stack has proven to be a perfect solution to meet the demands of today's web audiences.
There are many stacks used by developers, but the dominant two are MEAN (MongoDB, ExpressJS,AngularJS, NodeJS) and MERN (MongoDB, ExpressJS, ReactJS, NodeJS). So out of these two stack which is best for whom and for which projects, is a really a complex question. Therefore it is necessary to analyze the usesage of these two stacks in IT sectors. This research paper provides you a brief description of both stack i.e MEAN and MERN and also give a comparative analysis of both Stack.

KEYWORDS: Node JS, Angular JS, React JS, MEAN, MERN.

I. INTRODUCTION

The stacks used in web development are basically the response of software engineers to current demands. They have essentially adopted preexisting frameworks (including JavaScript) to make their lives easier. While there are many, MEAN and MERN are just two of the popular stacks that have evolved out of JavaScript. Both of these stacks are made up of open

source components and offer an end-to-end framework for building comprehensive web apps that enable browsers to connect with databases. The common theme between the two is JavaScript and this is also the key benefit of using either stack.
You can basically avoid any syntax errors or any confusion by just coding in one programming language, JavaScript. Another advantage of building your next web project with MEAN or MERN is the fact that you benefit from its enhanced flexibility [1].

II. The MEAN Stack

Software and Web Developers have been using Java Web Apps and stacks like LAMP (comprising Linux Operating System, the Apache Web Server, MySQL for database management and PHP for rendering view and control) to create dynamic web applications and services for a long time. However, a new development stack known as MEAN or MEAN stack has emerged a few years ago. The MEAN (MongoDB, Express, AngularJS, Node.js) stack was one of the early open-source stacks that epitomized this shift towards SPAs and adoption of NoSQL. AngularJS, a front-end framework based on the Model-View-Controller (MVC) design pattern, anchored this stack. MongoDB, a very popular NoSQL database was used for persistent data storage.MEAN is a collection of four tools that offer both client-side and server-side interactive applications. Mongodb provides the object database, Express.js provides the routing framework, Angular 2 is used for rendering the web-application and Node.js is a web server component and a JavaScript engine that is extremelys useful in injecting dependencies in Angular Js and Express [1]. All the constituents of this stack are managed by JavaScript which was initially a client side web programming language. The JavaScript language is lightweight and easy to write and the fact that it is used in all the components of the MEAN Stack makes it extremely easy to understand the data flow through the components from client to server.

A. Node.js

Node.js is an event-driven asynchronous JavaScript runtime which was primarily designed to make scalable network applications and APIs. Node Js is the best choice for real time and streaming application. Node Js was built on Chrome V8 Engine which is an open-source JavaScript engine developed on C++ and V8 implements the ECMA Script. Node.js uses an asynchronous event-driven [2], non-blocking I/O model that makes it lightweight, efficient and high-performance [3]. Although the Node is mentioned as the last tool of the MEAN stack, it is the most important one. Node.js' package ecosystem, npm (node package manager), is the largest ecosystem of open source libraries in the world which provides all the modules and dev Dependencies for Angular CLI along with all the Dependencies for Express. It acts as the dependency injector of the stack.

B. Express.js

Express.js is a minimal and flexible Node.js web application framework which provides a robust set of fundamental web application features without obscuring Node.js features. Express makes it easy to create APIs because of access to middleware. Middleware functions are functions which have access to the request and response objects along with a next function, which when invoked, executes the middleware succeeding the current middleware. Express.js provides a similar functionality to that of what spring offers to Java applications, that is, an easy to use web framework.

C. Angular.js

Angular is an open source web application framework, to create single page web application. It is a JavaScript framework which can be added to an HTML page or any web page, with a <script> tag. Angularjs uses ng-directives to extend HTML. The Angular Js directives are simply HTML attributes to which a prefix is attached- ng. Angular forms the front end of the MEAN stack. It implements the MVC pattern to differentiate between the client side and server side of an application and thus reduce the load on the server [4]. The initial launch version of Angular was Angular 1.x which used the concept of creating modules and registering them on HTML for responsiveness. The latest version of Angular which is Angular 2 uses the concept of creating Components that help in code reuse in any deployment target.

D. Mongodb

Mongodb is an open-source, document-based database. It is also called as a noSQL database program. It makes use of JSON like documents along with schemas. Because of the tremendous increase in the volume and variety of data over the past few years, there has been a need for a non-relational database. Mongodb fulfills these criteria, as it stores its data in a document-oriented, file structure. Each database can have multiple collections and data in those collections is stored in the form of array of JSON objects.

III. THE MERN STACK

The MERN stack is almost similar to MEAN stack except that React used in place of Angular. To understand MERN stack we have to understand firstly React Js. React was born not in the Facebook application that we all see, rather in Facebook’s Ads organization. Originally, they used a typical client-side MVC model to start with, which had all the regular two-way data binding and templates. Views would listen to changes on models, and they would respond to those changes by updating themselves. Soon, this got pretty hairy as the application became more and more complex. Cascading updates became difficult to maintain, because there would be subtle difference in the code to update the view, depending on the root cause of the update. That's when they started thinking of building something that's declarative rather than imperative [5].

A. React

A declarative efficient and flexible front end JavaScript library for user interface development. React Js can be used for both client and server side. React anchors the MERN stack. In some sense, this is the defining component of the MERN stack. React is an open-source JavaScript library maintained by Facebook that can be used for creating views rendered in HTML. Unlike Angular JS, React is not a framework. It is a library. Thus, it does not, by itself, dictate a framework pattern such as the MVC pattern. You use React to render a view (the V in MVC), but how to tie the rest of the application together is completely up to you.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 4, April 2018

IV. PROS AND CONS OF MEAN AND MERN

A. *PROS OF MEAN Stack*

- **JavaScript -** One of the most important benefits of all is that, it lets the developer write the entire code in JavaScript from client to server. This is like a blessing for the JavaScript developers who have invested their time and money in learning JavaScript for the client side tasks.
- It supports the **MVC (Model View Controller) architecture**.
- **Employability**: Employers around the world are on the lookout for engineers who are familiar with JavaScript-based technologies. By being proficient in the MEAN stack, not only does your future bode well, but you can also easily shift to other JavaScript technologies such as Backbone.JS without any problems
- **Lesser Costs and Faster Delivery:** If you are coding a whole project within the MEAN stack, you don't have to work in Silos anymore. This means better team communication, reduced impact on productivity and reduced friction, all of which combine together to bring down development costs while assuring faster delivery
- **Flexible and Better-designed Apps:** Not only are the apps developed with the MEAN stack incredibly optimized for performance, but are also better suited to multi-device responsive designs. Add to this the fact that coding with the MEAN stack gives more freedom to the designers, and you can be rest assured of a very flexible JavaScript stack
- **Speed:** When it comes to flat-out benchmark performance, Node.JS even beats such heavy-hitter like Apache! Being an event-driven architecture definitely helps, and so does the amount of optimization and development it is witnessing due to Google and Microsoft, etc.
- **OS Independence:** Many old favorites such as the LAMP stack, restricted the OS of choice for developers to Linux. The future of the MEAN stack, on the other hand, points to an OS-independent development framework which could work equally well on Windows, MacOS, and Linux.

B. *CONS OF MEAN Stack*

- MongoDB claims to be strongly consistent, but a lot of evidence recently has shown this not to be the case in certain scenarios (when network partitioning occurs, which can happen under heavy load). This means that you can potentially lose records that MongoDB has acknowledged as "successfully written"
- Relational databases: still the workhorse for a number of huge applications at companies like Facebook and Google. Granted, it takes more code to be productive, and they can much slower when you need to perform Joins, they still offer better overall functionality than most NoSQL solutions.
- Express.js poor isolation of the server from the business logic which prevented the reuse of services for purposes like batching operations (you don't want to have to go through the Express middleware chain for an internal request). Code instead of configuration, Express emphasized using code (imperative programming) instead using a more declarative model when wiring up middleware and defining routes.

C. *PROS OF MERN Stack*

- **JavaScript Everywhere:** The best part about MERN that I like is that there is a single language used everywhere. We use JavaScript for client-side code as well as server-side code. Even if you have database scripts (in MongoDB), you write them in JavaScript. So, the only language you need to know and be comfortable with is JavaScript.
- **JSON Everywhere:** When using the MERN stack, object representation is JSON (JavaScript Object Notation) everywhere – in the database, in the application server and on the client, and even on the wire.
- **Node.js Performance:** Due to its event driven architecture and non-blocking I/O, the claim is that Node.js is very fast and a resilient web server.

ISSN(Online): 2320-9801
ISSN (Print) : 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(A High Impact Factor, Monthly, Peer Reviewed Journal)

Website: www.ijircce.com

Vol. 6, Issue 4, April 2018

- **The npm Ecosystem:** I've already discussed about the huge number of npm packages available freely for everyone to use. Any problem that you face that you think others should have faced too, you'll find that there is an npm package for that. Even if it doesn't fit your needs exactly, you can fork it and make your own npm package.
- **Isomorphic:** SPAs used to have the problem that they were not SEO friendly. One had to use workarounds like running Phantom JS on the server to pseudo-generate HTML pages, or use Prerender.io services that did the same for you. These introduce an additional complexity.
- **It's not a Framework:** Not many people like or appreciate this, but I really like the fact that React is a library, not a framework.

V. COMPARATIVE STUDY OF MEAN VS MERN STACK

A. *Angular vs React*

While most developers can agree that the MEAN stack is a great option for modern app development, a large number are starting to advocate the use of React over angular. Developers are now starting to talk in terms of MEAN and MERN, the only difference being Angular or React. Angular is front end JavaScript framework whereas React is simply a JavaScript library. A frame is a structure for presenting code. It dictates a specific architecture for how your code is organized. Angular, for instance, brings an MVC architecture to front end development. It comes with additional helper functions and built in functionality for making http requests, etc [6]. At the end of the day, both are great options. While Angular has the backing of Google, React is a Facebook product. This means both products have great teams backing them and ample amounts of documentation, examples, etc.

B. *Performance*

React is arguably better for performance reasons. Angular 1 uses two way data binding and a digest cycle to constantly keep the view in sync with the underlying data model. This gets expensive when you have hundreds of variables to dynamically check and update. React uses unidirectional data flow to more efficiently check for state change. This plays better with larger data sets, where hundreds of thousands of records need to be rendered and updated. Although Angular 2 has introduced better state control, React is an easier and more intuitive way to handle change events.

C. *Architecture*

This goes back to the whole library vs framework discussion. While React makes UI rendering a breeze, it's just a library. It's up to you as the developer how you organize your code to work with underlying data models, etc. As a framework, Angular enforces an MVC like design, forcing developers to better organized their code. Although React is more flexible, it leaves more up to the developer as to how the app is organized. This can make your code harder to maintain.

D. *Third Party Libraries*

React relies heavily on other third party libraries to make things happen. A great example is with http requests. React doesn't have an out of the box solution for making http calls to a backend server. Angular has a built in $http service wrapper that makes http requests a breeze. Although libraries like Axios allow you to easily make requests in React, it requires more configuration.

E. *Angular 2?*

Angular 2 is still very new but it addresses some of the major pitfalls seen with Angular 1. It makes use of the shadow DOM (like React) to more efficiently control state change. It also allows for server side rendering and competes with React where Angular 1 cannot. The main headache with Angular 2 is Type Script. Type Script is a superset of JavaScript (developed by Microsoft) that essentially makes JavaScript more Java like in syntax. If you are a Java developer, then Angular 2 may be the perfect solution for you. Otherwise, the more strongly typed syntax can get very frustrating, very fast.

F. *COMPARISON TABLE*

S.N	**Attribute**	**MEAN**	**MERN**
1	DOM	Regular DOM	Virtual DOM
2	Learning Curve	Weak	Strong
3	Packaging	Weak	Strong
4	Abstraction	Weak	Strong
5	Debugging General	Good HTML / Bad JS	Good JS / Bad HTML
6	Debug Line NO	No	Yes
7	Unclosed Tag Mentioned?	No	Yes
S.N	**Attribute**	**MEAN**	**MERN**
8	Fails When?	Runtime	Compile-Time
9	Binding	Two Way	Uni-Directional
10	Templating	In HTML	In JSX Files
11	Component Model	Weak	Medium
12	Building Mobile?	Ionic Framework	React Native
13	MVC	Yes	View Layer Only
14	Rendering	Client Side	Server Side

VI. CONCLUSION

This paper gives you comparative results about MEAN vs. MERN Stack. This paper gives you a brief introduction of both Stacks. As we know MEAN and MERN stack differs from Angular and React. As it is a big problem for developers to choose a right stacks for application. So this study gives them an idea to choose.

REFERENCES

1. http://digi117.com/blog/choosing-the-right-stack-for-your-next-web-project-mean-vs-mern.html
2. MEAN.io. (2015) MEAN— Full-Stack JavaScript Using MongoDB, Express, AngularJS, and Node.js. [Online]. Available: http://mean.io/
3. S.Tilkov and S. Vinoski, ―Node.js: Using JavaScript to Build High-Performance Network Programs,‖ IEEE Internet Computing, vol. 14, no. 6, pp. 80 – 83, 2010. [Online] Available: http://doi.ieeecomputersociety.org/10.1109/MIC.2010.145
4. I.K. Chaniotis, K.-I. D. Kyriakou, and N. D. Tselikas, ―Is Node.js a viable option for building modern web applications? A performance evaluation study,‖ Computing, pp. 1– 22, 2014. [Online]. Available: http://dx.doi.org/10.1007/s00607-014-0394-9
5. A. Leff and J. T. Rayfield, ―Web-application development using the Model/View/Controller design pattern,‖Proceedings Fifth IEEE International Enterprise Distributed Object Computing Conference, pp. 118– 127, 2001.
6. Pro MERN Stack: Full Stack Web App Development with Mongo, Express, React - By Vasan Subramanian
7. MEAN vs MERN Stack : https://www.stackchief.com/blog/MEANvsMERNStack