The MEAN Stack

Mr. Ninaad Nirgudkar[1], Ms. Pooja Singh[2]

[1,2]Department of Information Technology, Bharati Vidyapeeth Institute of Managementt & Information Technology Navi Mumbai, Maharashtra,India

---

Abstract - *This paper examines the four components of MEAN stack (MongoDb, Express.js, Angular.js &Node.Js) and how they go well together, their benefits as a complete stack in web development. This paper also describes the work flow and server architecture in detail to understand the working of these four technologies employed in the MEAN stack web development. This paper mainly focuses on roles of these four technologies in MEAN stack and how they are popularly implemented in present times.*

***KeyWords*: MEAN Angula.js, Node.js, MongoDb, Express.js, LAMP, Web development.**

1. INTRODUCTION

To research and understand the working and intricacies of the modern technologies employed in the MEAN (MongoDb, ExpressJs, AngularJs, NodeJs) stack web development practices. The practicality of FULL-STACK development employs the MEAN Technologies and their ease-of-use characteristics for the modern developer. The four Technologies comprising the MEAN stack are MongoDb as database, Express as the Server Framework, Angular for front-end and Node.js as an event-driven I/O(input/output) server-side JavaScript environment. The main characteristic of the MEAN stack is that all four technologies are based on java script and JSON (JavaScript Object Notation) which is used to exchange data across these technologies saving potential time consumption of JSON encoding. Why MEAN? - Simplicity, uniformity and most of all, performance. The learning curve is sharp as it does not require a programmer to learn multiple programming languages, just JavaScript is enough.

2. LITERATURE SURVEY

In today's day and age, technology is growing at a rapid pace. With the new inventions of hardware devices and systems, it is natural for software development technologies to progress as well, actively replacing the old technologies. Due to the increasing number of electronic devices that use the Internet and their real-time nature of things, performance is key.
Traditionally, web development has been done using technologies such as JAVA servlets, PHP(Hypertext Preprocessor), ASP.NET(Active Server Pages), ruby etc. While these technologies are very popular and have extensive features and years of development, they have their own shortcomings compared to today's requirements with regards to performance. NodeJS, AngularJS, MongoDB and many more have been recently developed to fulfill today's need for a better alternative.

Google developed the V8 JavaScript engine which compiles and executes JavaScript source code, handles memory allocation and provides accurate garbage collection which is crucial for V8's high performance. NodeJS is a high performance JavaScript runtime environment which is built on V8.

A popular development style that makes use of the V8 engine in an effective manner is the MEAN stack. The MEAN stack has the advantage of Node's package ecosystem, npm(node package manager) which the largest ecosystem of open source libraries. Node uses JavaScript as its programming language for both server and client side. MongoDb is the database used to store the data the application needs to run, Angular is the front-end application running on the client side, Express is a lightweight HTTP(hypertext transfer protocol) server framework used to serve the Angular application and the resources it needs to run the app, and Node.js is the environment used to run Express.

3. THE MEAN STACK

Currently most popular and widely used open source web development stack is the LAMP (Linux, Apache, MySQL, and PHP) stack. Here Linux is the Operating system, Apache as the web server, MySQL as database and PHP as the programming language used for server side scripting. A newly emerging web development stack is the MEAN stack which uses MongoDb as database, Express as a flexible server framework that provides routing and handles request and response, Angular works at the client side.

3.1 Node.Js

NodeJs or just Node is the most important component of the MEAN stack. It provides the JavaScript development environment. It is built based on Google's V8 engine. Both Node and V8 are implemented in C and C++ for less memory consumption and faster performance. Node is based on Asynchronous I/O eventing model designed for developing scalable network applications. It fires callbacks on events, and each client event generates its own callback. If no work is to be done, Node is sleeping. While Node works on a single thread, it can serve many clients. Almost no function in Node directly performs I/O, they are handled by higher-order functions. Node presents the event-loop as a runtime construct, but unlike some other technologies, node does not have a blocking start-the-event-loop call. It simply enters the loop and exits upon completion similar to browser JavaScript. Node also has different modules that help take advantage of a multiprocessor environment such as creating child processes, sharing sockets etc.

3.2 Express.js

Express is a server side framework built in the NodeJs environment. It handles the client requests to the server and manages routing and HTTP methods such as GET, POST, PUT etc. Express configures Middlewares, which are basically functions that use the request, response objects and call the next middleware in the stack. It is the Middleware's responsibility to either end the request-response-cycle or pass the call next() to call the next middleware so the request isn't left hanging.

An express application is created by calling the express() exported by express e.g. app = express(). The app object is used to perform various operations and provide services by express. Express listens to a socket connection on a path or on a specified host and port number. Then using one of the METHOD() functions such as app. get() where app is the express application object and get() is the METHOD function, start the request-response-cycle of the appropriate middleware.

To configure middle wares the app. Route() returns an instance of a single route, which can be handles by HTTP methods and optionally middle wares. The app. render() is used to render HTML view files using a call back. Express uses template view engines to render views.

3.3 Angular.js

Angular.js is an open source JavaScript library developed and maintained by Google. It was developed with capabilities to handle the entire client side application and interaction. Its specifically used to develop a SPA (Single Page Application) that loads the entire web page on an initial request. Angular has the ability to perform client side routing. This helps lighten the load on the server by a considerable margin. Another specialty of Angular is that it is a MVW (Model View Whatever) architecture i.e. the developers are free to choose whichever way they want to implement Angular for their projects.
Angular uses Directives, which are HTML mark-ups which appear as html elements, attributes or even CSS classes. The directives are used to bind data, and DOM manipulations. The directive ng-app is used to define the Angular application. Views and models are managed by a controller. Further the application itself can be divided using modules that help code re-usability. Templates contain HTML and Angular elements that are rendered by the controllers and models used to display dynamic views to the user.

Angular can handle the entire client side routing. This is done using the directive ngRoutes. We can call controllers using the routes, and similarly render templates when necessary. The SPA ability of Angular is achieved through routing.

3.4 MongoDb

MongoDb is a document oriented NoSQL type of database. It stores data in JSON format. It has a dynamic schema and hence very popularly used to develop scalable applications. MongoDb does not require its users to know a traditional relational language such as SQL. Node has a package called mongoose that handles the interaction between MongoDb and the node, Data in mongo is stored in the form of objects or documents as they are refereed in mongo. BSON (binary encoded JSON) is the document format. BSON is extended from JSON and has a few extra data types that are not supported in JSON. Mongoose is used to perform CRUD (Create Read Update Delete) operations on MongoDb.

4. IMPLEMENTATION AND EASE-OF-USE

JavaScript started as a simple script that's meant to be run by the browser. Now, however, JavaScript is everywhere. It can be found running on smartphones, servers, Arduino, RaspberryPi and in many more technological developments. The Edge that JavaScript has over other languages is that, it is Non-blocking. A single non-blocking thread in JavaScript is more efficient than using threads in languages like java.
JSON is the common format used to exchange data between all 4 layers. Since JSON is native, no parsing is required at all. JSON is light-weight and easily consumed by JavaScript.

The most common and efficient way to use the MEAN stack is to use express to create a RESTful API, while angular handles the client side routes taking full advantage its SPA characteristics. Only when data from the database is required, will the application be required to make use of the API. This way most of the business logic can be applied and executed on the client side. Illustration can be found in fig.1

In the Express side of things, app handlers are used to handle the requests and give responses. These handlers receive the request and start request-response cycle with middlewares. User management, authentication, session management and the CRUD operations on MongoDb are handled by express.

Technologies can be hindered in their development if it is too hard to learn and the costs outweigh the benefits. However in the MEAN stack, these 4 technologies seamlessly integrate with each other e.g. express response object can directly be supplied to angular within any need for parsing.

MEAN.io and MEAN.js are popular Node packages that have all 4 technologies already pre-compiled and can be used directly without needing separate setup for them. This makes it especially easy for the developers since some part of the integration is already automated straight out-of-the-box.

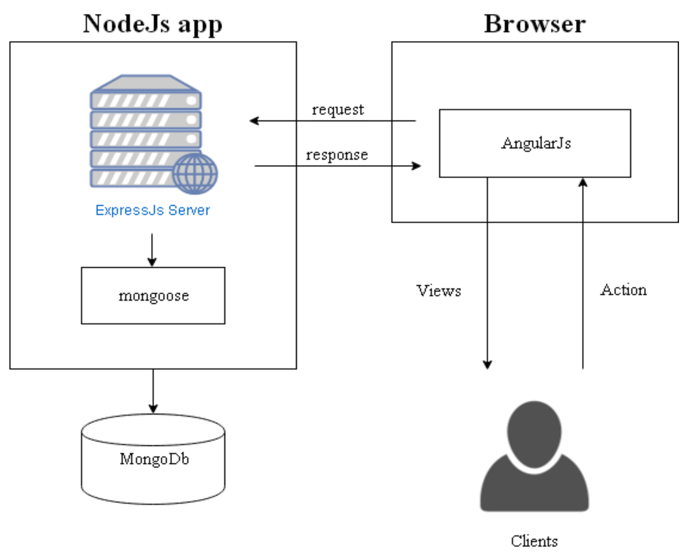

Fig-1: Workflow of MEAN stack

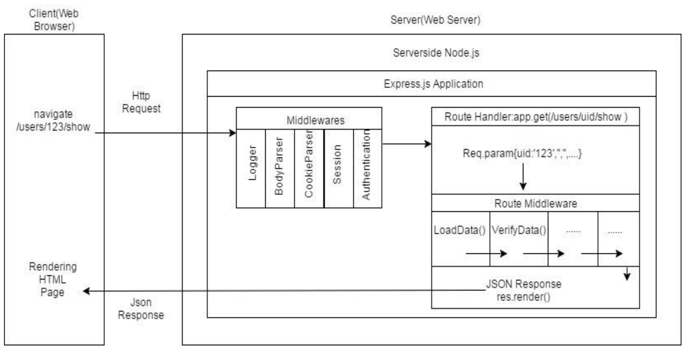

Fig-2: Server architecture of Express.js

5. LIMITATION

1. For the conversion of any project from LAMP to MEAN stack isn't that easy as it requires rewriting of the existing code in JavaScript.
2. MEAN stack is not suitable for the projects which have large source code because as team get to a certain size, it gets bit of hard to maintain and debug the code in JavaScript as it can get confusing which line of code is for front-end and which is for back-end.
3. MongoDb is not advisable for complex and bigger sets of big data. It may require a experienced person to work with it efficiently.
4. Node.js requires a dedicated application process to run the application which will get hard to deploy the entire MEAN application on shared hosting.
5. JSON is not as reliable as XML when transferring data between separate systems since there is no schema and namespace support.
6. Proper training is to be given to developers regarding mongo since it is very different from a traditional relational database. There are no joins and hence data distribution should take this into account. Many programmers fail to understand this and replicate the existing relational database into MongoDb and suffer for it.
7. Node.js runs on a single thread and eliminates problems of concurrency. However if concurrency is needed then the programmers themselves have to handle it.
8. The maturity level of Node and npm packages is low compared to old battle tested libraries of PHP, JAVA etc.. It is hard to find new reliable packages as anyone can post a package in the npm libraries. Many packages have reached a stable state through the years, however it can be challenging to find new reliable packages.

6. CONCLUSION

The popularity of JavaScript led to many technologies being developed around it. Today JavaScript is the pioneer in web development with technologies such as Angular, Node, React, and many more. While these technologies are less mature to their traditional counterparts, however surpasses them in many ways. MEAN stack in a basically a combination of such technologies that go well together and their applicability ranges from a regular web page to technologies developed in Internet of Things

REFERENCES

[1] https://nodejs.org/en/docs/ - NodeJs documentation.

[2] https://expressjs.com/ - ExpressJs documentation.

[3] https://docs.mongodb.com/ -MongoDb documentation.

[4] https://docs.angularjs.org/guide/concepts - Angular 1.x docs

[5] http://adrianmejia.com/blog/2014/09/28/angularjs-tutorial-for-beginners-with-nodejs-expressjs-and-mongodb/ - working with MEAN

[6] http://www.ijcter.com/papers/volume-2/issue-5/ease-of-mean-stack-in-web-development.pdf - LAMP and MEAN