

CHAPTER 1

INTRODUCTION

MEAN is a free and open-source JavaScript software stack for building dynamic web sites and web applications or we can say that MEAN stack is a collection of JavaScript based technologies used to develop web applications. MEAN is an acronym for MongoDB, ExpressJS, AngularJS and Node.js. From client to server to database, MEAN is full stack JavaScript.

- **M = MongoDB**, a popular database manager which implements a NoSQL structure.
- **E = Express.js**, a framework which supports and is used to host Node.js projects.
- **A = Angular.js**, yet another framework for building apps. It builds upon the classic html framework style and extends it to web apps
- **N = Node.js**, the crowning glory. This is a runtime environment, which runs server-side web applications, i.e. it works on the back-end, away from the user's eyes to fetch relevant data or perform operations on the same.

In a full stack web application, JavaScript is the consistent language and syntax. The MEAN stack is comprised of four main JavaScript oriented technologies. MongoDB is the database, Express is backend web framework, AngularJS is front-end web framework and Node.js is the back-end runtime environment. The reason why the stack is selected because developers will have improved view of greater picture by understanding the different areas and how they fit together. Frameworks and libraries are no longer limited in supporting user interfaces but also can be combined with other layers of applications. However, wrong architecture, low-level design and unbalance foundation in initial construction can dramatically affect the development process (Linnovate, 2014). Graph 1 explains the information of the MEAN Stack.

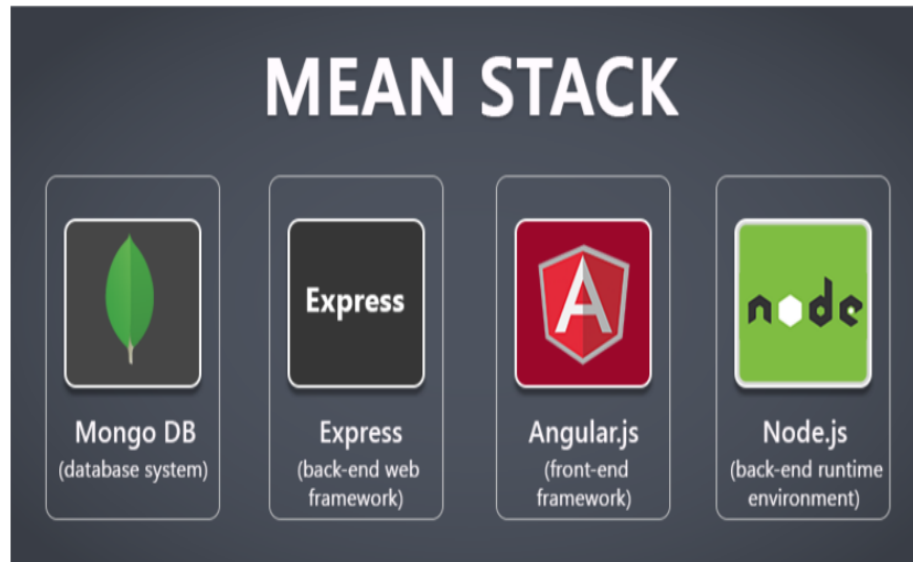


Fig 1.1 MEAN Stack Elements

MongoDB offers a more flexible, accommodating layer for storing data. Node.js provides a better nexus for running your server, while Express.js helps standardize how you build your websites. On the client, AngularJS provides a clean way of adding interactive functions and AJAX-driven rich components. Put them all together and they make a clean, coherent mechanism for moving data from user to disk farm and back again.

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.

Websites are one of the most effective ways to present and propagate information to people around the world. The web browser is a software application for retrieving, presenting and traversing information resources on the World Wide Web. More than ten years ago, a website was constructed with raw HTML document and served by simple HTTP request. The birth of CSS and JavaScript made web application become more and more convenient in displaying information. In this period, developing server side of a web application required developer know one of many programming languages, which are PHP, Java, C#. Otherwise, JavaScript is the primary scripting language to make reactive website. The development process is not an easy task for junior developers because they have to learn many programming languages along with client side markup language.

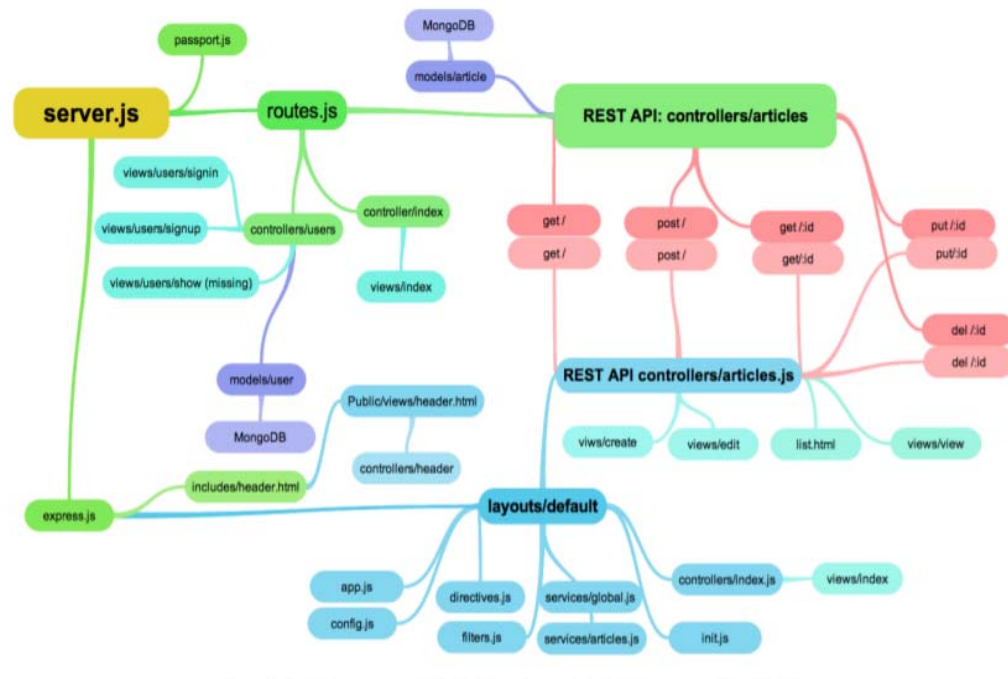


Fig 1.2 Diagram of MEAN Stack

Features of MEAN Stack

- 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.
- The MEAN components are open source; which means the stack gets updated regularly. In addition to it, it is easy and flexible to understand and use which helps the developers to customize as per your needs.
- Other advantages are the huge module library of node.js and the use of JSON is to transfer the data.
- Not a just startup, big players also move to Node.js: Walmart, PayPal, Yahoo, Netflix, Uber, LinkedIn.

Architecture of MEAN Stack

MEAN is a free and open-source JavaScript software stack for building dynamic web sites and web applications. MEAN is very simple and easy to use for both back end and front end in other technologies there are different languages used for front and back end but in written in one language for both server-side and client-side execution. Working of MEAN Stack is explained below in the figure properly.

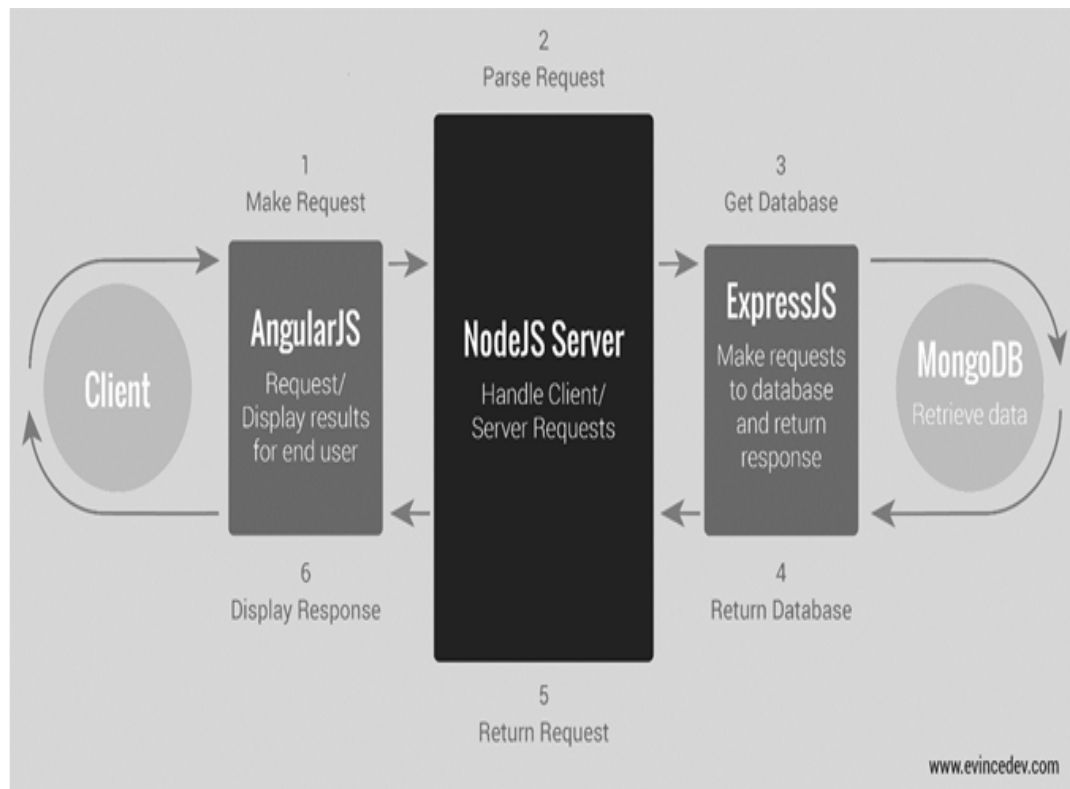


Fig 1.3 MEAN Stack Architecture

1. When client makes any request it is firstly processed by the AngularJS. AngularJS is a client side language in JavaScript.
2. After that the Request enters in phase 2 which is NodeJS. NodeJS is a server side language in JavaScript.
3. After that Request enter in the phase 3 which is ExpressJs it make request to the database.
4. After that MongoDB retrieve the data and return the response to the ExpressJs.
5. Then ExpressJs return response to the NodeJS and then NodeJS return it to the AngularJS to display the result.

CHAPTER 2

LITERATURE REVIEW

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.

CHAPTER 3

MODULES AND FUNCTIONALITIES

AngularJS (Client Side)

AngularJS is a structural framework for dynamic web apps. It lets you use HTML as your template language and lets you extend HTML's syntax to express your application's components clearly and succinctly. AngularJs data binding and dependency injection eliminates much of the code you currently have to write. And it all happens within the browser, making it an ideal partner with any server technology.

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.

1. AngularJS is a powerful JavaScript based development framework to create RICH Internet Application(RIA).

2. AngularJS provides developers options to write a client side application (using JavaScript) in a clean MVC (Model View Controller)
3. AngularJS is in open source, completely free, and used by thousands of developers around the world. It is licensed under the Apache License version 2.0.

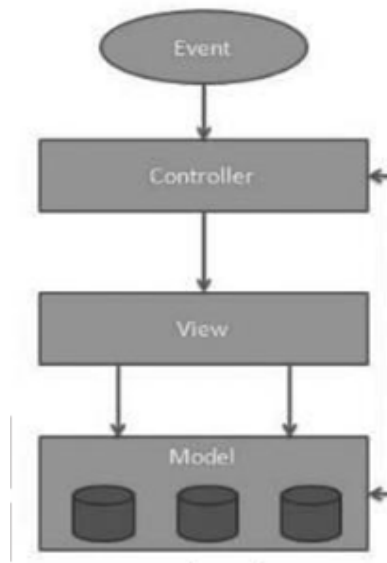


Fig 3.1 AngularJS Architecture

1) Model

- It is the lowest level of the pattern responsible for maintaining data.
- The model is responsible for managing application data. It responds to the request from view and to the instructions from the controller to update itself.

2) View

- It is responsible for displaying all or a portion of the data to the user.
- A presentation of data in a particular format, triggered by the controller's decision to present the data. They are script-based template systems such as JSP, ASP, PHP and very easy to integrate with AJAX technology.

3) Controller

- It is a software Code that controls the interactions between the Model and View.
- The controller responds to user input and performs interactions on the data model objects. The controller receives input, validates it, and then performs business operations that modify the state of the data model.

NodeJS (Server Side)

Node.js is a server-side platform built on Chrome's JavaScript runtime for easily building fast and scalable network applications. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient, perfect for data-intensive real-time applications that run across distributed devices.

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.

Node.js is a software platform allowing developers to create web servers and build web applications on it. Meanwhile PHP needs to run separate web server program such as Apache or Internet Information Services, Node.js server can be configured

when building an application. However, it is not limited in web projects. More phone apps and desktop applications are built on top of Node.js. (Node.js Foundation, 2012).

Node.js provides many utilities' packages in order to create a powerful web server that runs JavaScript on the server side, for example, asynchronous I/O, single-threaded or multi-threaded, sockets, and HTTP connections. When building web applications using other server-side programming languages, such as Java, PHP, and .NET, the server creates a new thread for each new connection. Assumed that there is a popular website receiving more than two million requests daily, developers probably need more servers or even to invest in more hardware. This approach works well as long as the server has enough hardware infrastructure to provide services to the customers. When the number of clients exceed the server's ability, it starts to slow down and the customers have to wait (Holmes, 2015).

The single threaded, non-blocking I/O changes how a request is handled by the server. Instead of creating a new thread for each connection, the web server receives client requests and places them in a queue, which is known as Event Queue. The Event Loop picks up one client request from the queue and starts process it. If the request does not include any blocking I/O operation, the server process everything, creates a response and sends it back to the client. In the case that the request can block the I/O operation, there is a different approach. An available thread from the thread pool will be picked up and assigned to the client request. That thread has then processed the blocking I/O, processed the request and created a response to send back to the client (Jörg, 2016). Node servers can support tens of thousands of simultaneous connections. It does not allocate one thread per connection model, but uses a model process per connection, creating only the memory that is required for each connection (Monteiro 2014, 32). Grap 3 explains single threaded application.

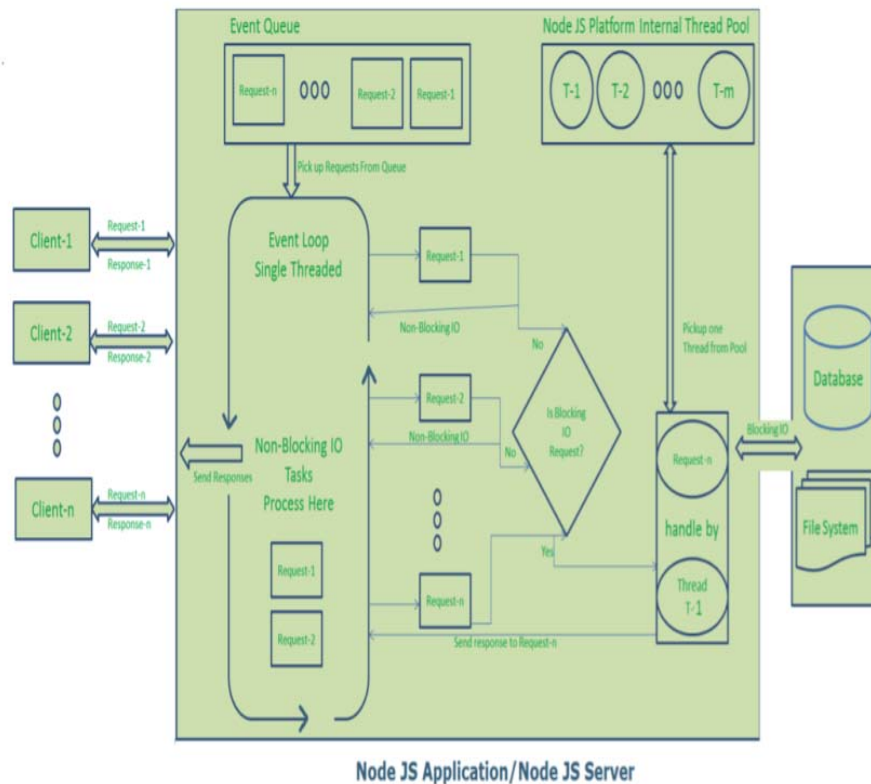


Fig 3.2 NodeJS Single Thread Event Model

1) Asynchronous and Event Driven

- All APIs of Node.js library are asynchronous, that is, non-blocking. It essentially means a Node.js based server never waits for an API to return data. The server moves to the next API after calling it and a notification mechanism of Events of Node.js helps the server to get a response from the previous API call.

2) Very Fast

- Being built on Google Chrome's V8 JavaScript Engine, Node.js library is very fast in code execution.

3) Single Threaded but Highly Scalable

- Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.

4) No Buffering

- Node.js uses a single threaded model with event looping. Event mechanism helps the server to respond in a non-blocking way and makes the server highly scalable as opposed to traditional servers which create limited threads to handle requests. Node.js uses a single threaded program and the same program can provide service to a much larger number of requests than traditional servers like Apache HTTP Server.

Threading

Node.js operates on a single thread, using non-blocking I/O calls, allowing it to support tens of thousands of concurrent connections without incurring the cost of thread context switching. The design of sharing a single thread between all the requests that uses the observer pattern is intended for building highly concurrent applications, where any function performing I/O must use a callback. In order to accommodate the single-threaded event loop, Node.js utilizes the libuv library that in turn uses a fixed-sized thread pool that is responsible for some of the non-blocking asynchronous I/O operations.

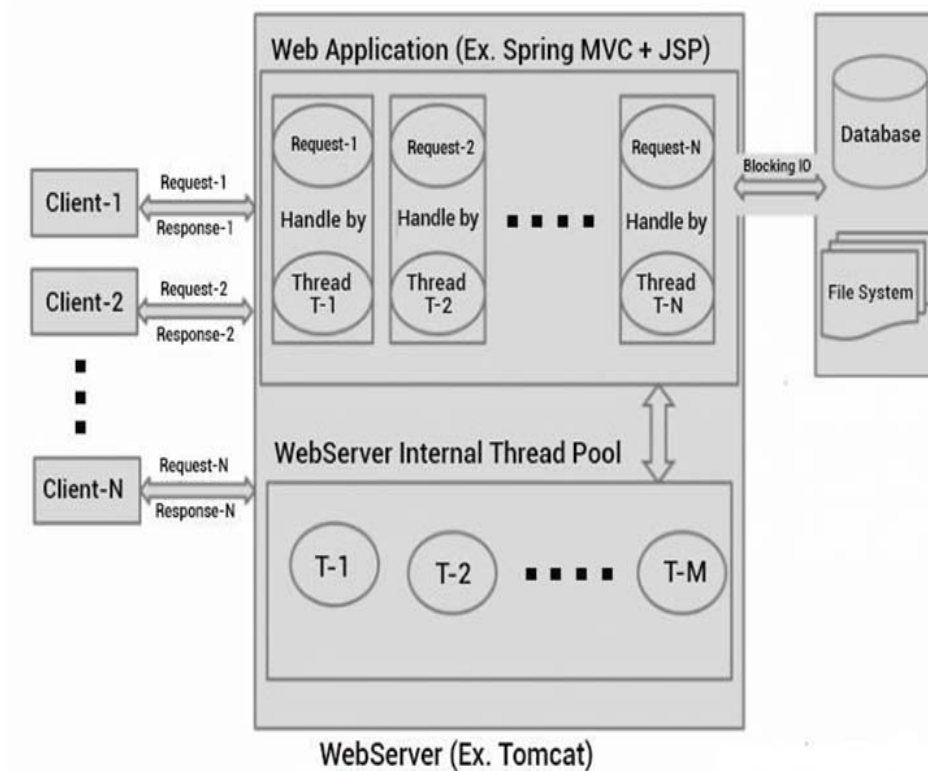


Fig 3.3 NodeJS Single Thread Architecture

ExpressJs

Express provides a minimal interface for us to build our applications. It is minimal, providing us the absolutely required tools to build our app and flexible, there are numerous modules available on npm for express, which can be directly plugged into express.

Unlike its competitors like Rails and Django, which have an opinionated way of building applications, express has no “best way” to do something. It is very flexible and pluggable.

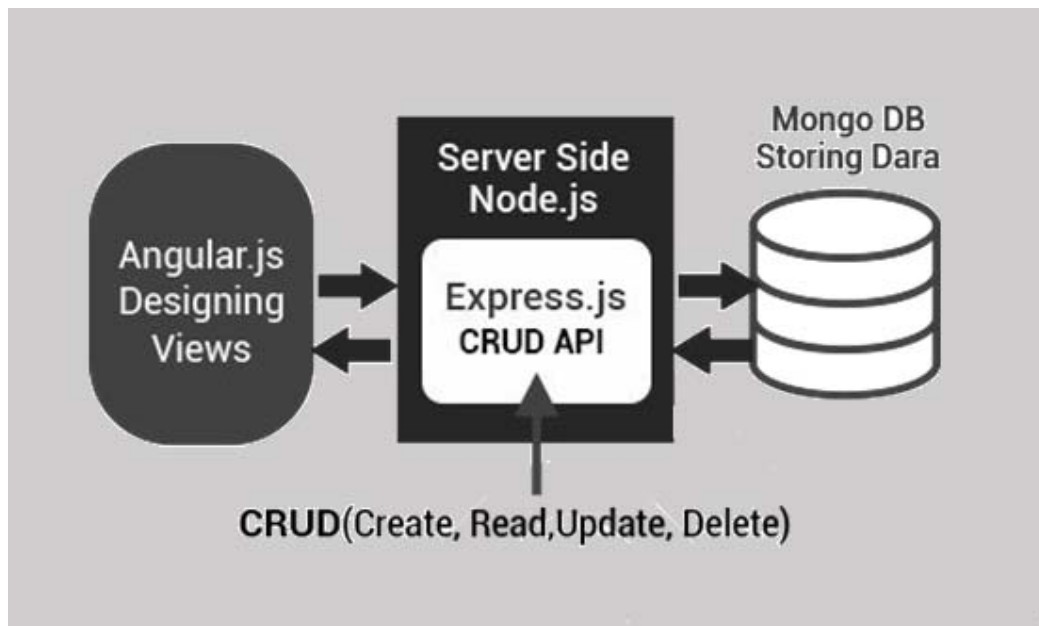


Fig 3.4 Architecture of ExpressJS

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 Middleware's, 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

Architecture Description

When the user sends a request through AngularJS then that request is firstly accessed by the NodeJS threading is done in the NodeJS and then it is sent to the ExpressJS to Create, Read, Update and Delete the API for the Request. ExpressJs host the website for the NodeJS. Both [NodeJS](#) and ExpressJS are server side languages. After CURD the API data is retrieved from the MongoDB and then send it to the User.

- Create (POST) – Make something
- Read (GET) – Get something
- Update (PUT) – Change something
- Delete (DELETE) – Remove something

MongoDB

MongoDB (from humongous) is a free and open-source, cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemas. MongoDB is the database for today's applications, enabling you to:

- Leverage data and technology to maximize competitive advantage
- Reduce risk for mission-critical deployments
- Accelerate time-to-value
- Dramatically lower total cost of ownership

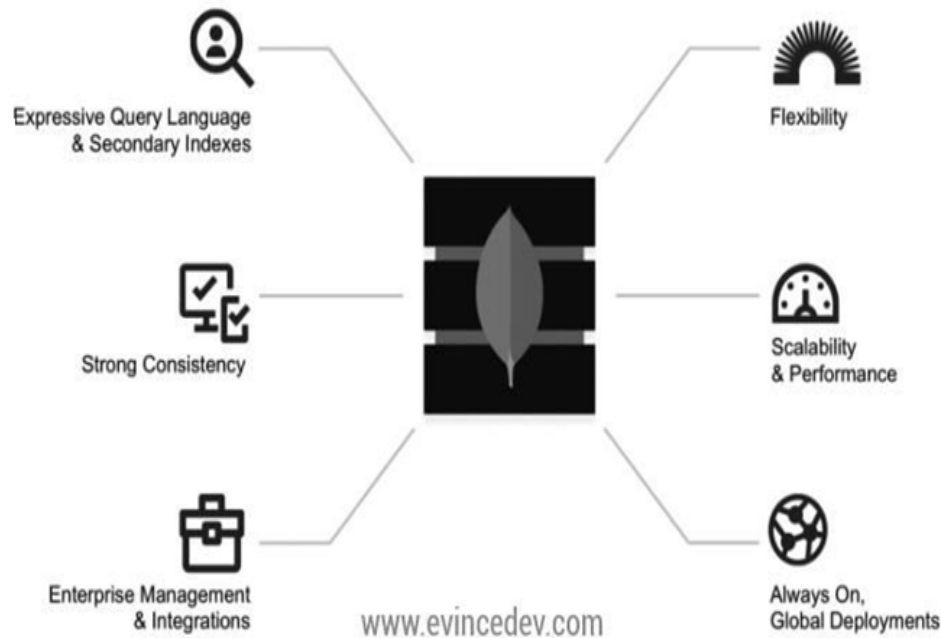


Fig 3.5 MongoDB Architecture

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.

Architecture Description

1) Expressive query language & secondary Indexes.

- Users should be able to access and manipulate their data in sophisticated ways to support both operational and analytical applications. Indexes play a critical role in providing efficient access to data, supported natively by the database rather than maintained in application code.

2) Strong consistency.

Applications should be able to immediately read what has been written to the database. It is much more complex to build applications around an eventually consistent model, imposing significant work on the developer, even for the most sophisticated engineering teams.

3) Enterprise Management and Integrations.

- Databases are just one piece of application infrastructure, and need to fit seamlessly into the enterprise IT stack. Organizations need a database that can be secured, monitored, automated, and integrated with their existing technology infrastructure, processes and staff, including operations teams, DBAs, and data analysts.

4) Flexible Data Model.

- NoSQL databases emerged to address the requirements for the data we see dominating modern applications. Whether document, graph, key-value, or wide-column, all of them offer a flexible data model, making it easy to store and combine data of any structure and allow dynamic modification of the schema without downtime or performance impact.

5) Scalability and Performance.

- NoSQL databases were all built with a focus on scalability, so they all include some form of sharding or partitioning. This allows the database to scale out on commodity hardware deployed on-premises or in the cloud, enabling almost unlimited growth with higher throughput and lower latency than relational databases.

6) Always-On Global Deployments.

- NoSQL databases are designed for highly available systems that provide a consistent, high-quality experience for users all over the world. They are designed to run across many nodes, including
- replication to automatically synchronize data across servers, racks, and data centers.

Features of MongoDB

- In MongoDB, data represents in a collection of JSON documents.
- MongoDB's querying is object-oriented, which MEANs you can pass MongoDB a document explaining what you are querying. MongoDB doesn't support joints, it supports multi-dimensional data types like other documents and arrays.
- MongoDB, you will only have one array of comments and one collection of posts within a post
- One of the best things about MongoDB is that you are not responsible for defining the schema

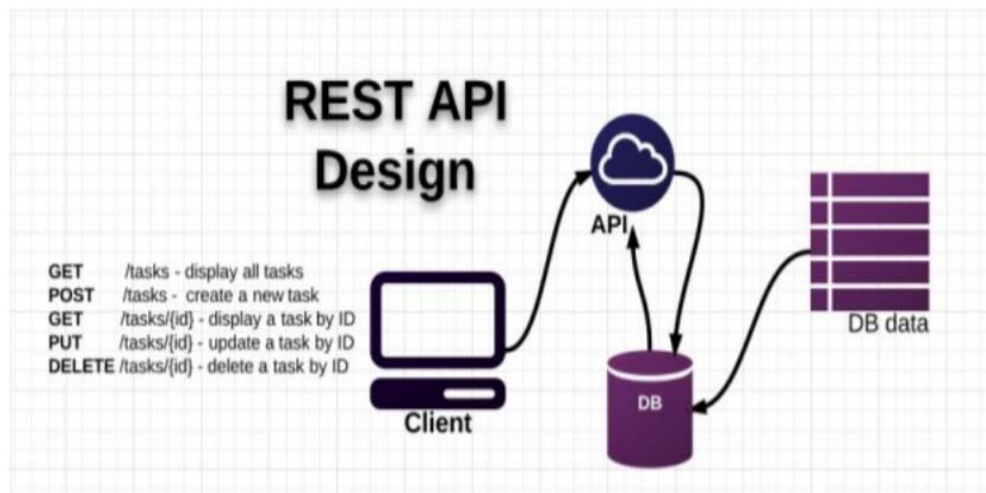
REST API and JSON

In legacy applications, XML-RPC is the method to transfer data via HTTP protocols. It is extremely complicated and uses a large number of bytes to encode objects. However, REST API that uses JSON become the universal connector for data on the internet. The JSON syntax is derived from JavaScript object notation syntax, but the JSON format is text only. Code for reading and generating JSON data can be written in any programming language .

REST API

REST stands for Representation State Transfer, which is a stateless connection between clients and servers. It is modern client-server architecture using the HTTP protocol to communicate and JSON as a formatting language. In a simple CRUD application as an example, REST API is used to create stateless interfaces for POST (create), PUT (update), GET (retrieve) and DELETE activities. The graph below

shows the example of typical REST API design (Hunter II, 2013). Graph 5 REST API design (Hunter II, 2013)



Graph 5 REST API design (Hunter II, 2013)

Fig 3.6 REST API DESIGN

JSON

JSON (JavaScript Object Notation) is a lightweight data-interchange format. No matter what knowledge of JavaScript developer has, it is easy to read and write. There is no problem for machines to parse and generate. Compared to XML, it is JavaScript-friendly. It also has the advantage of being generally easier to write by hand than XML. JSON has better JavaScript support and is much simpler to write (EcmaInternational, 2001).

CHAPTER-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 middleware's. 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-thebox.

CHAPTER-5

INSTALLATION

Before moving forward, Node.js, MongoDB and other necessary modules need to be installed. There must be a stable internet connection during installation and development process. MacOS is the main operating system and other Linux distros can follow the installation guide. Windows users can easily find a guide on the internet.

5.1 Node.js

As the main purpose is coding the web application on the Node.js runtime environment, it is crucial to install Node.js. Furthermore, npm is a tool to install modules. npm makes it easy for JavaScript developers to share the code that they have created to solve particular problems, and for other developers to reuse that code in their own applications (npm Inc, 2010).

5.1.1 Installing with GUI

Graph 8 displays the official page to download Node.js installation package. Depending on the local OS, users can download the necessary one. Developers should choose the most stable version in the LTS section. In the current tab, the page provides latest features for any operating system. However, they are not fully supported and can contain potential bugs. Downloading the most released version is not a wise move unless developers have been working for several years .

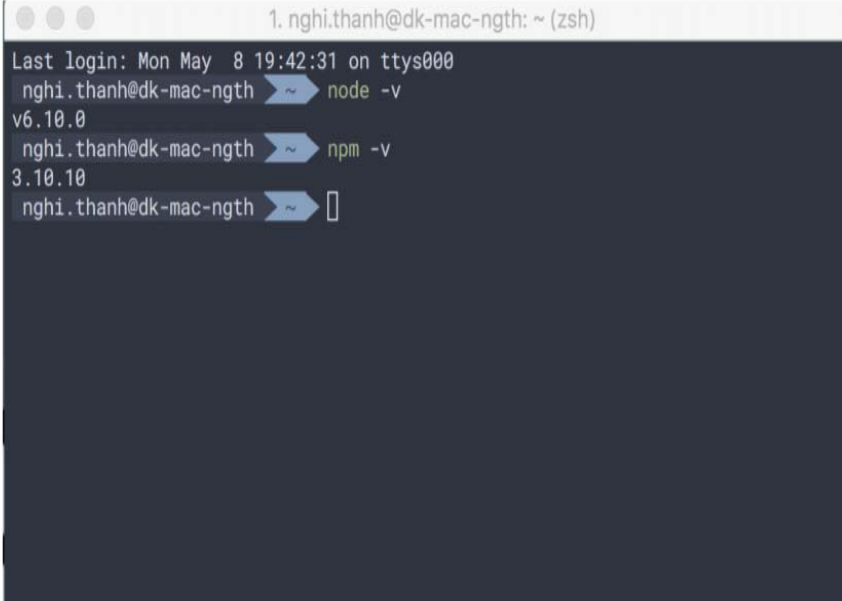


Graph 8 Node.js download page (Node.js Foundation, 2012)

Fig 5.1 NODE JS INSTALLATION

5.1.2 Using terminal

Homebrew, the package manager on MacOS, is needed to install Node.js. It is easy to find how to install Homebrew at the official web site. When Homebrew is downloaded, `brew install node` is the command that allows a user to download and install Node.js (McFarland, 2014). Graph 9 will show how to check if node is installed on the local machine. The two commands are used to check both node and npm version.

A terminal window titled '1. nghi.thanh@dk-mac-ngth: ~ (zsh)' with a dark background. It shows the output of two commands: 'node -v' resulting in 'v6.10.0' and 'npm -v' resulting in '3.10.10'. The prompt 'nghi.thanh@dk-mac-ngth ~' is visible at the bottom.

```
1. nghi.thanh@dk-mac-ngth: ~ (zsh)
Last login: Mon May  8 19:42:31 on ttys000
nghi.thanh@dk-mac-ngth ~$ node -v
v6.10.0
nghi.thanh@dk-mac-ngth ~$ npm -v
3.10.10
nghi.thanh@dk-mac-ngth ~$
```

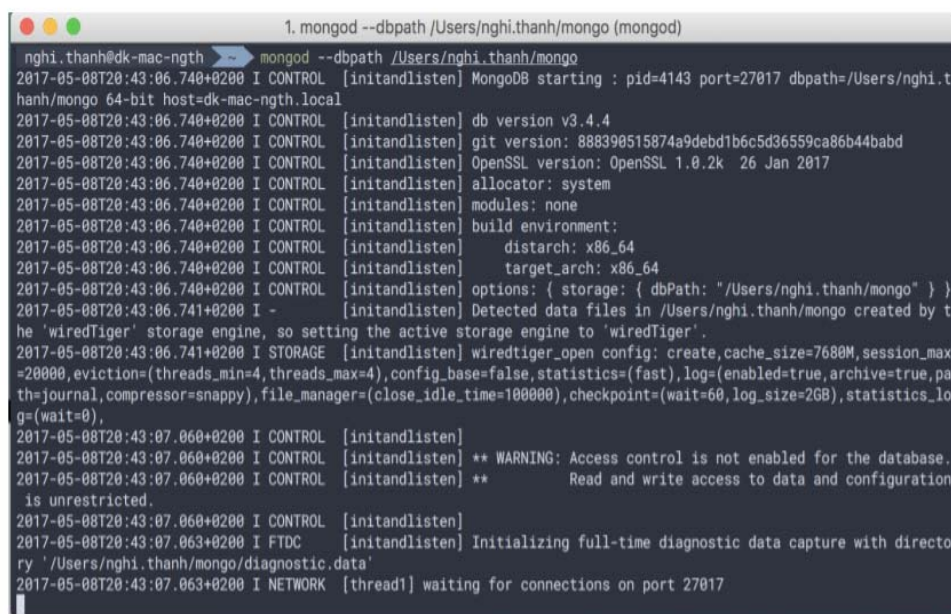
Graph 9 Checking Node.js on local machine

Fig 5.2 NODE JS ON LOCAL MACHINE

5.2 MongoDB

Following the demonstration with Node.js, MongoDB can be installed using GUI application or terminal commands. <https://docs.mongodb.com/manual/tutorial/install-mongodb-on-os-x/> displays the best way for favorite choice. After MongoDB is extracted, the following command shows how to connect to the mongod using command line. `mongod --dbpath pathToStoringDataDirectory` is the command to start

MongoDB in a local machine. The `pathToStoringDataDirectory` indicates where MongoDB can store a database in a directory. (MongoDB inc, 2008). If MongoDB is installed correctly, Graph 10 will show the result and connection port to MongoDB on local machine.



```

1. mongod --dbpath /Users/nghi.thanh/mongo (mongod)
nghi.thanh@dk-mac-ngth ~ % mongod --dbpath /Users/nghi.thanh/mongo
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten] MongoDB starting : pid=4143 port=27017 dbpath=/Users/nghi.t
hanh/mongo 64-bit host=dk-mac-ngth.local
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten] db version v3.4.4
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten] git version: 888390515874a9debd1b6c5d36559ca86b44babb
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten] OpenSSL version: OpenSSL 1.0.2k 26 Jan 2017
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten] allocator: system
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten] modules: none
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten] build environment:
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten]     distarch: x86_64
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten]     target_arch: x86_64
2017-05-08T20:43:06.740+0200 I CONTROL [initandlisten] options: { storage: { dbPath: "/Users/nghi.thanh/mongo" } }
2017-05-08T20:43:06.741+0200 I - [initandlisten] Detected data files in /Users/nghi.thanh/mongo created by t
he 'wiredTiger' storage engine, so setting the active storage engine to 'wiredTiger'.
2017-05-08T20:43:06.741+0200 I STORAGE [initandlisten] wiredtiger_open config: create,cache_size=7680M,session_max
=20000,eviction=(threads_min=4,threads_max=4),config_base=false,statistics=(fast),log=(enabled=true,archive=true,pa
th=journal,compressor=snappy),file_manager=(close_idle_time=100000),checkpoint=(wait=60,log_size=2GB),statistics_lo
g=(wait=0),
2017-05-08T20:43:07.060+0200 I CONTROL [initandlisten]
2017-05-08T20:43:07.060+0200 I CONTROL [initandlisten] ** WARNING: Access control is not enabled for the database.
2017-05-08T20:43:07.060+0200 I CONTROL [initandlisten] **      Read and write access to data and configuration
is unrestricted.
2017-05-08T20:43:07.060+0200 I CONTROL [initandlisten]
2017-05-08T20:43:07.063+0200 I FTDC [initandlisten] Initializing full-time diagnostic data capture with directo
ry '/Users/nghi.thanh/mongo/diagnostic.data'
2017-05-08T20:43:07.063+0200 I NETWORK [thread1] waiting for connections on port 27017

```

Graph 10 MongoDB successful connection on local machine.

Fig 5.3 MONGO DB CONNECTION

5.3 Utility Modules

This is the last stage of the installation process. There are three main modules which are extremely useful during the development process and can be used as command line. Bower is another package manager for front end development. Nodemon monitors any changes in source and restarts the server. Expressgenerator quickly creates an application skeleton. At this point, everything is ready to create a powerful MEAN stack application. `npm install -g express-generator bower nodemon` is the

command, which shows how to use npm to install node modules globally. After the command is executed, utility modules can be invoked as command lines.

5.3.1 Mongoose

When MongoDB is running and providing a port to connect as shown in Graph 10, developer can open the database and simply work with it by running the mongo command. It is the same as connecting to MySQL with terminal command line. A developer can create a new table and modify a data. Graph 11 shows work with MongoDB in a terminal on a local machine.

```

1. mongo (mongo)
X tail (tail)  %1 X ../aem-proje... %2 X mongod (mon... %3 X mongo (mongo) %4
Last login: Thu May 11 10:05:59 on ttys002
nghi.thanh@dk-mac-ngth ~ mongo
MongoDB shell version v3.4.4
connecting to: mongodb://127.0.0.1:27017
MongoDB server version: 3.4.4
Welcome to the MongoDB shell.
For interactive help, type "help".
For more comprehensive documentation, see
  http://docs.mongodb.org/
Questions? Try the support group
  http://groups.google.com/group/mongodb-user
Server has startup warnings:
2017-05-11T10:06:01.108+0200 I CONTROL [initandlisten]
2017-05-11T10:06:01.108+0200 I CONTROL [initandlisten] ** WARNING: Access control is not e
nabled for the database.
2017-05-11T10:06:01.108+0200 I CONTROL [initandlisten] **          Read and write access t
o data and configuration is unrestricted.
2017-05-11T10:06:01.108+0200 I CONTROL [initandlisten]
> show dbs
admin                0.000GB
admin-panel-github   0.000GB
local                 0.000GB
>

```

Graph 11 Working with MongoDB in terminal on local machine

Fig 5.4 WORKING WITH MongoDB TERMINAL

However, creating and removing data manually is not preferred at all. It takes time and contains syntax errors easily. That is why Mongoose Module is preferred.

Mongoose provides a straightforward, schema-based solution to model an application data. It includes built-in type casting, validation, query building, business logic hooks and more, out of the box (Automatic, 2011).

5.3.2 Async and Q

One of JavaScript's core features is a function as a first-class object. They can be stored in variables, passed as arguments to functions, created within functions, and returned from functions (Cunningham, 2014). According to Cunningham, passed functions can be executed inside a function. This feature is called callback. Developers sometimes encounter numerous levels of callback functions which looks like Graph 12.

```
fs.readdir(source, function (err, files) {
  if (err) {
    console.log('Error finding files: ' + err)
  } else {
    files.forEach(function (filename, fileIndex) {
      console.log(filename)
      gm(source + filename).size(function (err, values) {
        if (err) {
          console.log('Error identifying file size: ' + err)
        } else {
          console.log(filename + ' : ' + values)
          aspect = (values.width / values.height)
          widths.forEach(function (width, widthIndex) {
            height = Math.round(width / aspect)
            console.log('resizing ' + filename + 'to ' + height + 'x' + height)
            this.resize(width, height).write(dest + 'w' + width + '_' + filename, function(err) {
              if (err) console.log('Error writing file: ' + err)
            })
          }).bind(this)
        }
      })
    })
  }
})
```

Graph 12 Callback example (Ogden, 2012)

Fig 5.5 CALLBACK EXAMPLE

CHAPTER 6

ADVANTAGES AND DISADVANTAGES

ADVANTAGES

1.)An easy switch between client and server

The main reason why developers enjoy using MEAN is that they need to write code only in one language – Javascript, because it works for both server side and client side requirements.

If you are an expert in Javascript, then you can manage the entire project with MEAN stack, and deploy the app directly on the server, without having to go through the standalone server.

There is no need for Apache or LAMP stack because you already have Node.js acting like a web server.

2.) It is open source

The technologies in stack are open source, making them available and free. This gives you easier access to public repositories and libraries making the development process easy and less costly.

Top app developers have posted their answers to common queries, and even if you have any doubts, posting it there would invoke answers from the experts.

3.)Uses JSON

As MongoDB is a component based relational database, you can save documents in the JSON format (Javascript Object Notation). This is however, limited to small to intermediate level companies. This is probably why developers prefer this tech stack for various stages of app development.

4.)It's cost effective

MEAN is a cost-effective technology for businesses. The obvious reason is of course, you do not have to hire many experts to do a single task – a full stack Javascript

developer would suffice. This means you have a lot of time and money at your disposal, with the opportunity to hire only specialists.

5.)Allows for real time demonstrations of apps

At a time when the trend is to have real time demo app open to viewers/ subscribers, this technology would be a major help as it allows for quick and real time changes to your app even during app development.

6.)A respected time-saver

If you are plagued by a deadline, then the way to go is MEAN. The main reason is that you don't have to create any modules from scratch because you already have Node.js with its huge collection of module libraries.

Additionally, the automatic testing feature sends a notification when a particular feature has an error, so you can correct it before moving on.

7.)Supports MVC architecture

The productivity of the development team stays strong, thanks to the MEAN stack's capability of complying with the MVC or Model-View-Controller architecture. This gives them the freedom from dealing with a variety of programming languages.

8.) A universal programming language

Javascript, being a programming language, gives the developer team the advantage of adapting to a new comer if he happens to join in mid-project. It eases the new developer into the work schedule, especially if he is comfortable with the language.

It is also helpful when your web application evolves over time and needs additions and updates. This makes scalability possible, complete with the benefit of quick deployment too.

9.)A proven technology

Being a proven technology stack, you will pretty much get solutions on the way if you happen to get any along the way. Companies have already taken advantage of the benefits of Node.js, right from small startups to big tech giants.

10.) Highly flexible

You can test your app even during the development process, and host it on the cloud. And if you want to add new information any time in the future, just a new field to add in the form, and it's done.

DISADVANTAGE

1.) Could potentially lose records

The claim is that MongoDB is strongly consistent, but sometimes, this could change. When network partitioning occurs, especially in heavy load scenarios, there is a chance that you could lose records that has been successfully written by MongoDB.

This doesn't have to happen all the time, but chances are present.

2.) Poor isolation of server from business logic

Express.js has poor isolation of server from business logic and this prevents the reuse of certain services like batching operations. You have to go through the Express middle chain for internal jobs, and this could be bothersome.

3.) Can't beat the power of relational databases yet

MEAN stack, when compared to relational databases doesn't provide the same level of functionality. Relational databases is till the first choice for many big names like Google and Facebook, as it is more reliable and stable.

CHAPTER 7

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 is 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.

BIBLIOGRAPHY

- [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/angularjstutorial-for-beginners-with-nodejs-expressjs-andmongodb/> - working with MEAN
- [6] <http://www.ijcter.com/papers/volume-2/issue-5/easeof-mean-stack-in-web-development.pdf> - LAMP and MEAN