iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 9 **Issue:** VIII **Month of publication:** August 2021

DOI: https://doi.org/10.22214/ijraset.2021.37464

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.429
Volume 9 Issue VIII Aug 2021- Available at www.ijraset.com

Challenges with Implementation of Node.Js

Harsh Bhardwaj[1], Mr. Rajendra Khatana[2]
[1, 2]*Raffles University*

Abstract: Node.js gave rise to Full Stack Developers, who can now manage both the server and client sides on their own. Due to its event-driven, non-blocking, and asynchronous techniques, Node.js is fast and dependable for high file and heavy network traffic applications, where developers may also maintain a complete project on single pages (SPA) and can utilise for IOT. The study's findings are based on a survey and a review of the literature on Node.js implementation areas and obstacles. Finally, shall make recommendations on how to improve in order to overcome the issues.

I. INTRODUCTION

Web-based applications are becoming more popular as they become easier to create, manage, and secure. They are also conveniently accessible to clients, do not require additional installs in most circumstances, and are instantly adaptable. A web application is derived from a web-based system that has additional functionality to execute an organization's business logic. These programmes are entirely web-based, rather than requiring the installation of a separate application on the operating system. Web apps include Google Docs, web-based retail businesses, Google Maps, and web-based email programmes.

There are two types of developers in the Web development industry: front-end developers and back-end developers. Front-end developers must be familiar with HTML, CSS, and a programming language, such as JavaScript, in order to add effects and other features to the front-end. They create the web site's presentation and effects that are displayed to clients by converting the designer's design. Back-end Developers create the business logic that powers any online application. Back-end developers do tasks such as adding and retrieving news highlights to and from web applications, sending email from web-based forms, and validating visitors' or clients' credentials. Backend developers must be fluent in languages such as PHP,.NET, Java, and others. Back-end developers should also be familiar with databases such as My SQL, Oracle, and SQL Server, or they should employ or delegate the flow to a database administrator. A database administrator will look after the database server and assure its seamless operation.

Full stack developers are kings of all trades, and they do it all. Back-end developers are usually recommended to have front-end developer capabilities, and vice versa, putting an added strain on them to learn multiple skill sets. So it is clear that in order to become a full stack developer, a developer must have expertise in three types of languages: client side languages such as Java script along with HTML and CSS, server side languages such as PHP,.NET, Java, Ruby, and so on, and database expertise such as SQL Server, MySQL Server, and Oracle.

Node.js eliminates the need to learn multiple languages at the same time in order to become a Full Stack developer, whereas a front-end developer who is competent in JavaScript only needs to learn some more Server Side Functions to become a Back-end developer. Aside from the fact that Node.js is free, it is also utilised by thousands of developers all around the world. While the web application requires performance and scalability, threading and events have been used in the past, either independently or in combination. However, the increasing nature of everyday internet traffic necessitates improved and new methods to improve the concurrent nature of the service. One option is Node.js (also known as Node), which is lightweight and meets the demands through an event-driven and non-blocking I/O architecture and server-side JavaScript. Chrome's V8 JavaScript engine serves as the foundation for Node.js, as the Node.js JavaScript runtime is based on it. V8, a high-performance JavaScript engine written in C++, is available as open source from Google. V8 was utilised by Google in the Chrome browser, and it may run standalone as well as be integrated in C++ programmes. It can run on various versions of Windows, Mac OS, and Linux. Because Node.js is event-based rather than thread-based, it can scale to millions of connections concurrently while employing an event loop within a single thread and without the expense of numerous threads. Node.js performs I/O operations asynchronously and is sometimes confused with AJAX. While Node.js and AJAX are not the same thing.

A. Advantages of Node.js Versus Competing Frameworks

Node.js is designed from the bottom up to handle asynchronous I/O since it is built with JavaScript, and JavaScript is designed as an event loop. In client-side JavaScript, the on click event for a button is an event loop. While other environments offer this feature, it is done through the use of third-party libraries or is not developed from the ground up for the same purpose as Node.js, hence it is often slow or lags and does not belong as a standard feature to them. Some examples include Event Machine, which was built for

©IJRASET: All Rights are Reserved

Ruby, Twisted, which was introduced for Python and is available since Python 2 onwards, and Apache MINA, which is also known as the "Networking Socket Library" and is another example of providing event-driven and asynchronous limited to APIs only.
In the same way, Apache AsyncWeb is created with Apache MINA and Perl's Any Event. Similarly, one advantage of Node.js over others will be its ability to handle many requests while acting as a client to third-party services by operating only a single thread. Other languages, in this sense, will halt processing until the remote server responds to their original request, necessitating multiple threading for executions. In comparison, everything of what you will use in Node is asynchronous because writing non-asynchronous code in it will be difficult. Furthermore, Node.js will never require you to buffer data before outputting it, whereas others, such as Event Machine, would require you to buffer data in numerous circumstances.
Being server side JavaScript, another notable advantage of Node.js over others is that a developer will only be required to have knowledge and experience in a single language, namely JavaScript, regardless of whether he is building client side scripts or server side scripts. The developer is not obliged to switch his brain cycles from one language on the client side to another language on the server side. Furthermore, because Node.js is young, it has the advantage of avoiding mistakes that other languages have made in the past, such as the mistake of backward compatibility. According to statistics, around 47 percent of online surfers expect a page to load within 2 seconds, and a 3 second delay reduces consumer satisfaction by 16 percent. Node.js takes the lead here since its interpreter is smaller and faster than that of other languages such as PHP. Unlike other languages, where every application initiation will follow cycles consuming processes such as loading setups, followed by database connectivity, acquiring essential information, and lastly rendering the markup language, here the server side programmes are always maintained ON. Node.js, on the other hand, minimises these steps by keeping a programme always ON.

B. Disadvantages of Node.js Versus Competing Frameworks

Because it employs an event-driven/callback paradigm, Node.js code grows quickly while also being difficult to debug. The lack of ready hosting for Node.js environments is currently a major disadvantage. Complex themes in the JavaScript language, such as prototype inheritance, anonymous functions, and callbacks, make the language difficult to learn and, as a result, it is best learned after mastering another simple language. Node.js is still a young language, and as a result, professional programmers are hesitant to join. Another issue is that because the system is single-threaded, other requests are halted if the CPU is occupied for even a fraction of a second. As a result, developers are also compelled to think in asynchronous terms, which is difficult to adapt to.
The study's scope is to make struggles for researching Node.js implementations and trying to catch on how it might be made easy to implement for new arrivals in particular. It also attempts to investigate the hurdles to Node.js, such as why it is still not extensively used and why newcomers to Node.js are few. Efforts will be made to link concepts with real-world experiences, rather than relying solely on theory. The study will look at any concerns with Node.js implementations and uptake and make recommendations to address those concerns. This study will pique the reader's curiosity in adopting Node.js by revealing the characteristics and capabilities of Node.js.
The findings of this study will assist developers in determining causes and strategies for fast adopting Node.js, allowing Node.js to become as popular in the market as other top languages such as.Net Languages and PHP. This paper is organised as follows: section 2 will explore the review of literature, section 3 will detail the materials and methods utilised for this study, section 4 will portray the results, and section 5 will offer discussion and future work.

II. LITERATURE REVIEW

This section explains the existing research on Node.js as a server side language and areas where Node.js is implemented effectively, as well as some general overview on the same such as the History of Node, and certain application areas of Node.js and whether it is efficient or not.

A. Concerning Node.js and JavaScript

JavaScript's history can be traced back to the early days of the World Wide Web, when it began to play an important role in making the front ends of websites interactive. In the late 1990s, JavaScript debuted its concepts in AJAX, which is now utilised to bring real-time behaviour to online sites. Until recently, JavaScript was thought to be a client-side scripting language with little to do with server-side programming. However, the introduction of server side JavaScript, such as Node.js, modified the paradigm, and now JavaScript is not only a client side scripting computer language, but it can also be executed on the server side. Node.js is a major rival in the JavaScript server-side era. It is vital to note that Node.js is not the same as JavaScript; while JavaScript is undoubtedly the foundation of Node, Node.js is built on top of JavaScript and hence uses the same language.

©IJRASET: All Rights are Reserved

B. Node.js History and Evolution

Ryan Dahl was inspired to design Node.js after seeing a progress bar showing file upload at Flickr (a subsidiary of Yahoo dealing with photo galleries), where the browser was repeatedly querying the server how much of a file was being uploaded. In March of 2009, Node.js was given a name for the first time, and the package manager for it was introduced in October of the same year, with an early preview of npm (the Node.js Package Manager) being released. Later that year, in November, the creator Ryan Dahl gave the first Node.js session at JSConf 2009, where he discussed Node.js in depth. He discussed how Node.js is event-based and operates on callbacks, and how any I/O function, such as receiving input from disc, network, or any process, should use a callback, and the audience applauded for his wonderful project. The Express framework was introduced to Node.js in 2010. Node.js was not accessible for the Windows environment until July 2011, when Microsoft teamed with Joyent to make it available, and support for older versions of Windows Server was enabled. Later, near the end of 2014, some Node.js team fans boycotted Node.js and established their own fork of Node.js, branding it io.js or iojs. They boycotted because they were dissatisfied with Joyent's authority over the project. The next year, both appear to merge again as they both voted to form a neutral Node.js Foundation, and finally in the same year, in September 2015, they merged back and their integrated community made great progress in emerging a joined codebase.

C. The Node.js Platform's Architecture

Web servers gained event-driven programming with the release of Node.js. This made the web server fast and in a commonly used language, JavaScript, and it is also the reason that the whole web development community quickly gained access to Node.js. When designing highly scalable servers with Node.js, developers are not needed to employ threading. Node.js has a simple event-driven architecture that involves the execution of callback functions at the conclusion of a job or the development of an error. Ryan Dahl founded Node.js on the premise that other programming languages make it difficult to programme things to work concurrently.

D. Adoption of Node.js

PayPal, LinkedIn, Medium, and Netflix are just a few of the companies that use Node.js. Facebook adopted Node.js, which proved to be incredibly proficient for them, and they then embraced it for production. Microsoft made a significant improvement by including Node.js into the developer stack, mostly because developers can now use it on the Azure platform as well as others, and it is no longer limited to only obsolete operating systems. Walmart implemented end-to-end JavaScript using Node.js, believing that Node.js will assist them in fronting their services that they provide all over the world. They stated that they adopted Node.js not only because it introduces new ways of thinking about how to design flawless software, but also because it is a terrific method to represent existing things in new ways. JavaScript is the most popular language, according to GitHub statistics.

E. Application Of Node.js

Node.js was compared against PHP/Nginx for Performance and Scalability, with Node.js outperforming PHP/Nginx in both areas. The two researchers created a web application based on the Dijkstra Algorithm and used load generators to mimic the load of concurrent user queries. The author of a project report published by the University of Notre Dame ran testing between Node.js and Ruby's Event Machine and Apache's threading model, which assessed request time over the number of cores. Node.js had once again outperformed the other two, particularly when the number of cores was raised. The performance of Node.js-based DPWS – Devices Profile for Web Services (termed as Node. DPWS) was assessed and compared to other DPWS tools. The researcher determined that the Node.js-based DPWS was simple to use and light in weight. Node. DPWS had defeated even the most prominent alternative in the world of IoT. (Internet of Things). Node.js is also essential in the GIS sector, as evidenced by a work published in the Journal of Korea Spatial Information Society that used Node.js in Web GIS. They used Node map and found in their research that Internet GIS has a promising future when combined with Node.js. Another analytical study found that Node.js can be used to create complicated real-time applications that can be supplied to millions of client connections. There is a web service called MAGI that is used in graphic processing unit (GPU) infrastructures for quick data analysis of Micro RNA-seq. While MAGI is built on Node.js, it addresses the limitations of existing similar programmes, such as their inability to handle huge files and their cumbersome error-prone stages. Also, MAGI assisted in identifying the delay in downstream analysis because the others are time consuming, as well as resolving the issue of others not being able to offer statistical tests.

1) *Single Page Applications:* Single Page Apps (SPA) are web-based applications that do not require a page to reload while in use. While SPAs have a long history and are based on Java, Flash, and JavaScript. JavaScript, unlike the previous ones, does not require any third-party client plugins. That is why Node.js, which is based on JavaScript, is successful in competing with others in SPAs.

©IJRASET: All Rights are Reserved

2) *NodeOS:* Node OS is a Java Script-based operating system. Its packages are maintained by the Node.js package manager, npm, therefore every Node.js package is a NodeOS package. While there are approximately 300,000 packages in npm.
3) *Poison Tap:* Poison Tap uses Node.js to develop backdoors into even the most secure systems. Even if the USB is locked, it only has to be hooked into an operating computer. It then instals continuous backdoors that run even if the USB is transferred later, allowing the intruder to access the router as well as the target network and web browser cookies. When we asked the creator, Samy Kamkar, why he chose Node.js for Poison Tap, he replied via email, "The browser-based code must be in JavaScript, thus it made logical to me to retain the backend in the same language."
4) *Node.js in IoT:* Node.js is used by developers and researchers for IOT because JavaScript is fast and familiar to a large number of web developers because they use it with HTML5 to programme front end User Interfaces. Another important point is that JavaScript is best for embedded devices because its nature is to support asynchronous and event-driven functions. Furthermore, the programming paradigm used by Node.js is an excellent fit for embedded devices as well as servers, and domain specialists have already adopted Node.js for the purpose of IOT. Microsoft has also included Node.js in its developer resources for IOT development.

III. RESULTS AND FINDINGS

The Literature Review found that Node.js can be useful and should be used everywhere huge files are processed or a substantial network demand is required. Some of the outcomes in the same situation are detailed below. Because of Node.js, a developer can simply become a Full Stack Developer, which means that he does not need to work with distinct colleagues for server side development and database development. Employers can also save money by using Node.js because a single developer will be in charge of executing all tasks on both the server and client sides. The introduction of Node.js also brought for the easy implementation of Single Page Applications (SPA), where web applications written in it are faster since they require fewer server resources and have fewer callbacks to the server, making webpages more interactive and user friendly. The analysis of the literature also indicated that the subject of Internet GIS has a promising future when combined with Node.js. It's also worth noting that Node.js has been shown to be faster than other programming languages when it comes to graphics processing. As there are advantages to utilising Node.js, there are also risks of misusing it. One such example is the publication of PoisonTap USB, which creates backdoors to a computer and the network via any running machine, even if it is password protected. Because of the rising use of JavaScript among developers, such backdoor software has become a part of their daily lives.

Because the study is also based on quantitative research, a survey was conducted, and the findings of that survey are listed below. The online poll was distributed to a number of professionals via LinkedIn, Facebook, and personal networking. There were a total of 93 answers. Because they were developers and this study is relevant to the developer community, 80 of their responses were useful for this study. 16.2 percent of the 80 developers (respondents) contacted within a month were unaware of Node.js. The remaining 83.8 percent of developers completed the remaining survey questions, totaling 67 in number.

1) In terms of learning difficulty, respondents indicate that learning Node.js is difficult, with only 31.3 percent stating that learning Node.js was not difficult for them. 23.9 percent saw learning as a struggle, while 44.8 percent saw learning as a minor challenge.
2) Regarding the database difficulty, the poll indicated that because developers are experienced with SQL databases, they find it difficult to adopt NoSQL databases as rapidly as possible, with only 31.3 percent saying that using and learning NoSQL databases is not a challenge.
3) Regarding the Event-Driven feature challenge, the survey concluded that they felt difficulty with Event-Driven feature of Node.js as 40.3% did not considered it as a challenge, the rest25.4% felt it a challenge to some extent only, and the rest 34.3% said they felt it as a challenge.
4) In terms of the Non-blocking feature issue, the survey determined that this feature is a challenge for developers. Only 41.8 percent did not consider it a problem, while 31.3 percent consider it a difficulty, and 26.9 percent consider it a struggle to some level.
5) Concerning the asynchronous feature challenge, the poll found that only 43.3 percent of developers think it is amazing, 38.85 think it is a struggle, and 17.9 percent think it is a challenge to some extent only.
6) Concerning the challenge of being familiar with other programming languages, the survey concluded that developers with experience in other languages also enjoy Node.js. According to the survey results, 50.7 percent of respondents who use other programming languages do not consider Node.js adoption to be a challenge, while the remaining 17.9 percent of respondents from other programming languages consider Node.js adoption to be a challenge. If they come from another programming language environment, just 31.3 percent see it as a difficulty.

©IJRASET: All Rights are Reserved

7) Concerning the difficulty of setting a server for Node.js, 61.2 percent stated that they did not find it difficult. Only 25.4 percent saw it as a challenge, while 13.4 percent saw it as a problem only to some extent.

8) Concerning the problem of organisational decisions to embrace Node.js, survey results show that 47.8 percent of respondents believe their organization's decision is the reason for delaying Node.js implementation. 37.3 percent disagreed, while 14.9 percent are unsure about this question.

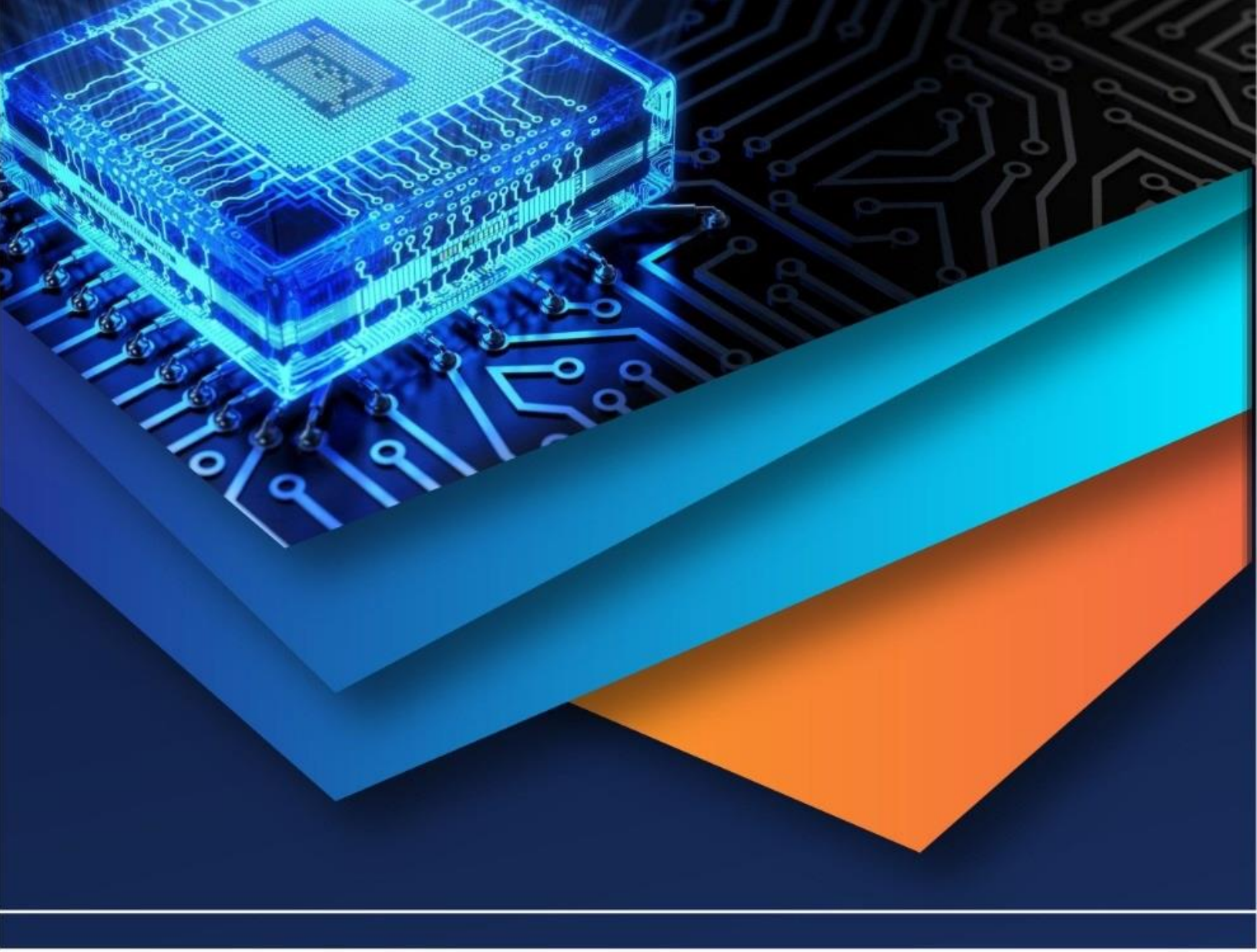
9) The difficulty of gaining market awareness is also an issue in using Node.js. When asked about the market awareness issue, 44.8 percent of online survey respondents felt it was a challenge, while 19.4 percent said it was a challenge to some level. While only 35.8 percent of those polled claimed it is not a challenge.

According to the results of the poll, the fact that the **same consistent language is used on both the servers and the clients** is a highly valued feature of Node.js (by approximately 50% of developers). Other aspects like as event-driven, non-blocking, and the use of JavaScript are also preferred by approximately 46 percent of developers. And some of them (34.3 percent) liked its ability to make itself fit best in IOT.

REFERENCES

[1] "The benefits of web-based applications," [Online]. Available: http://www.magicwebsolutions.co.uk/blog /the-benefits-of-web-basedapplications.htm. [Accessed 25 November 2016].
[2] Web Application Basics, Pearson Higher Education
[3] F. Bridge, "What Types of Developers Are There?," tree house, 24 June 2016. [Online]. Available: http://blog.teamtreehouse.com/what-types-ofdeveloper-are-there. [Accessed 25 November 2016].
[4] M. Wales, "Front-End vs Back-End vs Full Stack Developers," Udacity, 08 December 2014. [Online]. Available:http://blog.udacity.com/2014/12/front-endvs-back-end-vs-full-stack-web-developers.html. [Accessed 25 November 2016].
[5] J. Long, "I Don't Speak Your Language: Frontend vs. Backend," tree house, 25 September 2012. [Online]. Available: http:// blog.teamtreehouse.com/ i-dont-speak-your-language-frontend-vs-backend. [Accessed 25 November 2016].
[6] A. Mardan, "PHP vs. Node.js," Programming Weblog, [Online]. Available: http://webapplog.com/ php-vs-node-js/. [Accessed 28 January 2016].
[7] J. Kaplan-Moss, "Quora," [Online]. Available: https://www.quora.com/What-are-the-benefits-ofdeveloping-in-Node-js-versus-Python. [Accessed 29 June 2016].
[8] "Node.js Tutorial," tutorials point, [Online]. Available: https://www.tutorialspoint.com/nodejs/index.htm. [Accessed 25 November 2016].
[9] N. Chhetri, "A Comparative Analysis of Node.js (Server-Side JavaScript)," Culminating Projects in Computer Science and Information Technology., p. 5, 2016.
[10] R. R. McCune, "Node.js Paradigms and Benchmarks," 2011.
[11] Node.js, "Home page of Node.js," Joyent, 2016. [Online].Available:https://nodejs.org/en/. [Accessed 01 May 2016].
[12] G. Developers, "Chrome V8 | Google Developers," Google, [Online]. Available: https:// developers. google.com/v8/. [Accessed 27 May 2016].
[13] eventmachine, "GitHub, Inc," [Online]. Available: https://github.com/eventmachine/eventmachine.[Ac -cessed 30 June 2016].
[14] Twisted Matrix Labs, "Twisted Matrix Labs," [Online]. Available: http://twistedmatrix.com/trac/. [Accessed 30 June 2016].
[15] The Apache Software Foundation, "Apache MINA," [Online]. Available: http://mina.apache.org/. [Accessed 30 June 2016].
[16] The Apache Software Foundation, "Apache MINA," [Online]. Available: http:// mina.apache.org/ asyncweb-project/index.html. [Accessed 30 June 2016].
[17] Ryan Dahl: Original Node.js presentation. [Film]. Youtube, 2012.
[18] "How Loading Time Effects Your Bottom Line," Kissmetrics Blog, [Online]. Available: https://blog. kissmetrics.com/loading-time/. [Accessed 25 November 2016].
[19] C. Buckler, "Site Point Smack Down: PHP vs Node.js," Site Point, [Online]. Available: http://www. sitepoint.com/sitepoint-smackdown-php-vs-nodejs/. [Accessed 28 January 2016].
[20] Firehose, "Firehose," Firehose Project, [Online]. Available:http://blog.thefirehoseproject.com/posts/n odejs-vs-rails/. [Accessed 30 June 2016].

©IJRASET: All Rights are Reserved



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Call : 08813907089 (24*7 Support on Whatsapp)