PERFORMANCE OPTIMIZATION OF WEB-BASED APPLICATION

Abubakar Bello Bada

Department of Computer Science,
Federal University Birnin-Kebbi, Nigeria

ABSTRACT

Web-based applications have become the new way for people to interact, be it work or business. Users always expect a fast response from the applications. The performance of web-based applications has always been a critical non-functional requirement. A better performing web-based system leads to better user experience which leads to repeated visits and in turn more revenues. Performance problems can bring all kinds of undesired consequences like bad reputation, loss of revenue and low productivity especially in an e-commerce system. Recent advancement leads to mobile device proliferation, increase in traffic volumes and richer content, which all increase application delivery challenges. Users not only expect fast application response time, but also expect higher throughput.

In this work an e-commerce system is created, a performance tuning up was done on it to reduce the number of HTTP requests made and also reduce the size and number of pages and images in the system. Other ways of performance optimization were also considered. The system's performance was tested before and after performance tuning up, a load test was also done for 3000, 5000 and 10000 concurrent users. The results were compared to see the effect of the performance optimization techniques used.

Keywords: Web-based application, Performance, HTTP request, E-commerce, Optimization techniques, Load test.

INTRODUCTION

The current advancement of technologies used in developing a web-based application puts a lot of emphasis on the look and feel which makes the front-end of the application more complex [9]. Nowadays, bad user experience not only occur due to database, servers and other infrastructures but also due to time it takes to load the page and display its content on the user's screen (response time) [1] [7].

Most modern applications are created using layered architecture, mainly three-layered architecture: Presentation layer, logic layer and database layer [2]. Performance optimization of a web-based application can be done at two levels: Front-end and Back-end [8]. Optimizing front-end performance without losing the look and feel of the web-based application is very important for a good user experience. Another importance of this optimization is that search engines like Google rank websites based on their web page speed [2].

PERFORMANCE OF A WEB-BASED APPLICATION

Performance of a web application is the ability of that application to respond to a request within an appreciable time [7]. In other words, Performance of a web application refers to its ability to complete requests and furnish information rapidly and accurately despite high multi-user interaction or constrained hardware resources [5]. Thus, it is explained by response time (time it takes the system to respond to a user's request) and throughput (number of requests handled by the system in a given period of time) [4]. The performance of a web application depends on many factors including the network, the end systems, the application and most importantly the front-end [3].

Performance of a web application is determined by both front-end and back-end. Web application performance at the client-side is primarily driven by the amount of data transmitted over the wire, while at the server-side, programming language choice and platform, implementation complexity and configuration are the performance indicators [12]. Nowadays more emphasis is given to front-end when it comes to performance optimization because even though bottlenecks at the server-side (back-end) can make the application useless, their presence at the front-end is what determines user experience [1], [11], [12]. Optimizing the back-end performance by 50% can improve the application performance of the system up to 10% only, but overall application performance can be improved by 40% or even more by only reducing the front-end loading time to half [1], [11]. Moreover, optimization of front-end is simpler and has more cost-benefit compared to back-end optimization, as such this work put more emphasis on front-end optimization.

OPTIMIZING THE PERFORMANCE

Performance optimization is a technique(s) deployed to increase the ability of a system to respond to user's request within the shortest possible time and/or execute a good number of requests in a given time [14], [16]. As explained earlier, performance is determined by response time and throughput [1]. In reducing the response time (performance optimization at the front-end), ways that will minimize the total number of resource requests (HTTP requests) to the server and reduction in round-trip time for additional resources must be implemented as they delay the loading of a system [1] [15] [14]. Sizes of the pages and images in a system contribute to its slow response time, because pages and images with big size take longer time before been fetched from the server, hence their size reduction will effectively decrease the response time [19] [6] [7]. Processing of a large number of user requests concurrently with minimal response time (throughput) improves the performance at the back-end [16].

Reduction to the barest minimum of the number of pages that make up the system will also decrease the loading time as it reduces the number of HTTP requests sent to the server [10] [12].

Ajax was used as the technology to create the system used in this work because it provides opportunities to create and execute some functionalities at the client-side without sending any request to the server, this reduces the round-trip time [17] [3].

There are many ways in which performance of a web-based application can be tuned up to achieve a better response time while preserving the look, feel and responsiveness of the application [6] [2]. It also allows HTTP requests to be sent asynchronously with Javascript code, without reloading the entire HTML page sources. Ajax is very good in improving the front-end performance of a web application [17]. The performance optimization techniques studied and implemented after creating an e-commerce system for this work include:

1. Minification

This is the removal of all unnecessary characters such as spaces, newline and comments from the source code without changing the functionalities. It can be applied to JavaScript, CSS, HTML and PHP.

In implementing this method, all the CSS, HTML, PHP and JavaScript files are minified using htmlcompressor.com and jscompress.com, open source compression sites, which results in reduction of their sizes by a greater percentage. This helps in the reduction of the size of the files which in turn reduces the time to transmit them from the server to the client-side.

2. Reformat Images

This is a process of formatting all the images to a format(s) that will help in saving bandwidth and reduce response time. The format of choice here is JPEG, because the compression technique behind it reduces the size of the image without losing the image quality by removing all the unnecessary data. This helps in reducing the size of the images and make their transmission from the server to the user-side faster.

3. Content Delivery Network

This is a system of network that delivers web content to a user based on geographical location and the origin of webpage. User response time can be reduced to a greater extent by just distributing static web contents on various locations. Instead of including jQuery and JavaScript libraries in the code, CloudFlare content delivery network was used to deliver the JavaScript and jQuery libraries in this work, this helps in reducing the number of traffic to the server and also reduces the response time.

4. Consolidation of Scripts

This is combining scripts (CSS and JavaScript) into common files that can be shared by multiple pages. Because each script and stylesheet have to be fetched by HTTP request to the server, this adds performance overhead as it creates network traffic between the server and client. Caching the scripts by the browser will reduce the number of HTTP requests needed to render the page thereby improving the performance. Caching of the scripts is enabled by using external sheets, this enables them to be requested separately from the main document.

The Browsers often use progressive rendering i.e. rendering whatever content is available as soon as possible, misplaced references to style sheets delay the rendering of a web page by forcing the browser to defer rendering of the entire document until those references have been resolved. It is therefore pertinent to put references to style sheets at the top of the HTML document to allow for proper progressive rendering and, consequently, improved application performance. Because of this external CSS and JavaScript were used, this helps to reduce the number of HTTP requests to the server thereby decreasing the Round-Trip Time.

5. Compression of pages

Compression technologies help reduce the size of Web pages thereby decreasing the response time. This is mainly done by adding processing steps to compress on the server and decompress on the browser. Though old browsers do not support the decompression but most new browsers do.

GZIP Compression, a simple and effective way to save bandwidth and speed up system was enabled on the server. This decreases the size of the pages at the server level thereby making them faster to be transmitted to the browser. The pages are decompressed at the browser level.

Files that are already compressed like PDF and images don't need to be compressed again as this may affect CPU utilization and also increase their size.

6. Defer Parsing of JavaScript

The JavaScript files are put in the end of the body tag instead of the head so that the page is fully downloaded before downloading the JavaScript files. Putting the files at the beginning of the page will delay the page rendering as the JavaScript files have to be fully downloaded before continuing with the page rendering. This increases the response time of the system.

7. Cache-Control Header

Using browser cache is another way of optimizing performance. When a user visits a page for the first time it makes a lot of HTTP requests to download all the page resources (images, stylesheets, scripts), using browser cache, the resources are cached locally therefore don't have to be downloaded on subsequent visits as they have been cached before. A user can perform a conditional GET request by supplying the If-Modified-Since header. In response to a conditional GET, if the resource has not changed, the application server may return a 304 Not Modified response with nobody; this reduces the amount of transmitted data. An application server may choose to supply the Expires and/or Cache-Control response headers with responses; these are used to signal the client that a resource should only be re-retrieved after a particular date or period of time has passed. Proper use of the above headers may result in a significant reduction in the number of HTTP requests. Thus caching should be used, whenever possible, to improve application performance

8. Avoid Redirects

Redirects are achieved by using HTTP status codes of 3xx e.g. 301 (Moved permanently) and 302 (Found) status codes. Redirect is an indication that a user needs to take some additional actions to complete the request. The major problem with redirects is that they slow down the page load time. These redirects normally take place on various stages like when backslash (/) is not put at the end of the URL. Backslash redirects can be fixed in Apache by using Alias.

Avoiding redirects will increase the performance of a web application.

9. Reduce the Domain Name System (DNS) lookups

The DNS maps domain names to IP addresses. DNS normally takes 20-120 milliseconds to lookup the IP address for given domains and browser can't download anything from this hostname unless DNS lookup is completed.

Although DNS lookups are cached for better performance and it's information is placed on the operating system's DNS cache, most browsers have their own cache as well. Some browsers like internet explorer, cache the DNS lookup for 30 minutes by default and in a situation like this operating system cache has no use when DNS record exists in browser cache.

Number of DNS lookups will be equal to the unique hosts in the web page in case of empty browser cache. So, reducing the number of unique hostnames will reduce the page load time.

10. Implementation

Server-side code quality, architectural complexity and selection of third-party libraries and/or modules can have a significant effect on a web application performance. Hence proper selection of algorithms, architectural models and libraries, use of well-established coding idioms, optimization of database queries, effective use of application frameworks and efficient modularization, among others, are very important in optimizing the performance.

Three-tier architecture was used in the development of this system and all the libraries used e.g. jQuery library were referenced at the end of the page. Content delivery was used for the libraries as explained earlier.

11. Configuration

Proper configuration of application and database servers is very important for good performance of a web application. An application server (and the application running in it) must often handle requests from multiple concurrent clients (this helps with the throughput). Thus, the configuration of thread pools, database connection pools, memory management (e.g. garbage collection) etc. can have a substantial effect on performance. Proper configuration of database properties is also important.

RESULTS

For a web-based application to have a better performance, a lower response time and higher throughput must be achieved. An online shopping system (e-commerce) was created for the purpose of this work, this is because there are a lot of images on this kind of system and high number of users can be on it at the same time.

Web page test, a free online service which provides the website front-end speed test facility, was used to perform a performance test before and after optimization with the results as follows:

						Document Complete			Fully Loaded		
Load Time	First Byte	Start Render	Visually Complete	Speed Index	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In
6.731s	0.614s	6.163s	6.200s	6200	143	6.731s	16	291 KB	6.731s	16	291 KB

Figure 1: Performance test before optimization.

					Document Complete			Fully Loaded		
Load Time	First Byte	Start Render	Speed Index	DOM Elements	Time	Requests	Bytes In	Time	Requests	Bytes In
2.031s	0.663s	1.786s	1967	157	2.031s	17	290 KB	2.352s	18	294 KB

Figure 2: Performance test after optimization

Figures above show that it took approximately 6.7 seconds to load the system before performance was optimized and approximately 2 seconds after optimization. Implementing those performance optimization techniques drastically reduced the response time thereby making the loading of the system faster which in turn enhanced the performance.

Since in this study we are comparing the performance of the system before and after the implementation of the performance optimization techniques, we can safely say that the result is very good as it enhanced the performance of the system.

A load test was also performed on the system as performance can be greatly influenced by the number of concurrent users of the system. This test was done to check the performance of the system under heavy traffic. The test was conducted with Blitz, a very interesting tool that allows load testing on web applications using virtual users spread across the globe. Blitz is not an open source tool.

A first test with a duration of one minute was carried out with users going from 1 to 3000 concurrently using the system, the average response time of 3.05 seconds was achieved as shown in Figure 3.

A second test with users going from 1 to 5000 was conducted over a period of one minute and an average response time of 4.96 seconds was recorded as shown in Figure 4.

During the third test with concurrent users of 10000 within the same period of time, an average response time of 5.13 seconds was recorded as shown in Figure 5. Connection timeout error was recorded in this test as the users start approaching 7000.

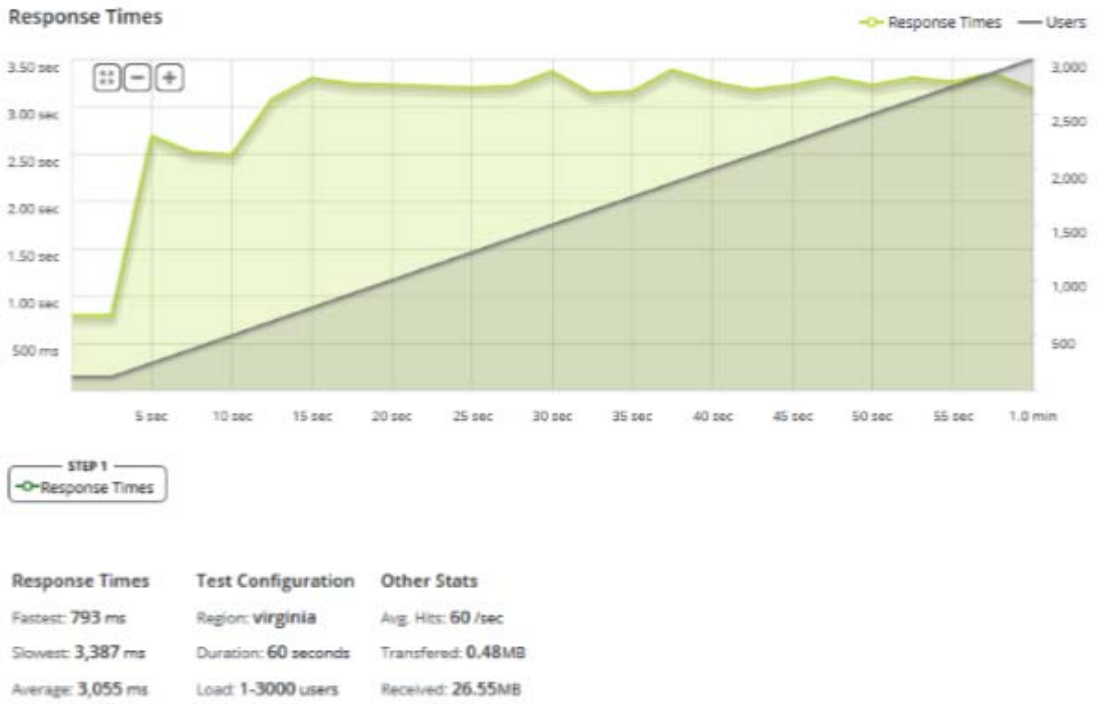

Figure 3: Load Test with 1 to 3000 users

Figure 4: Load Test with 1 to 5000 users

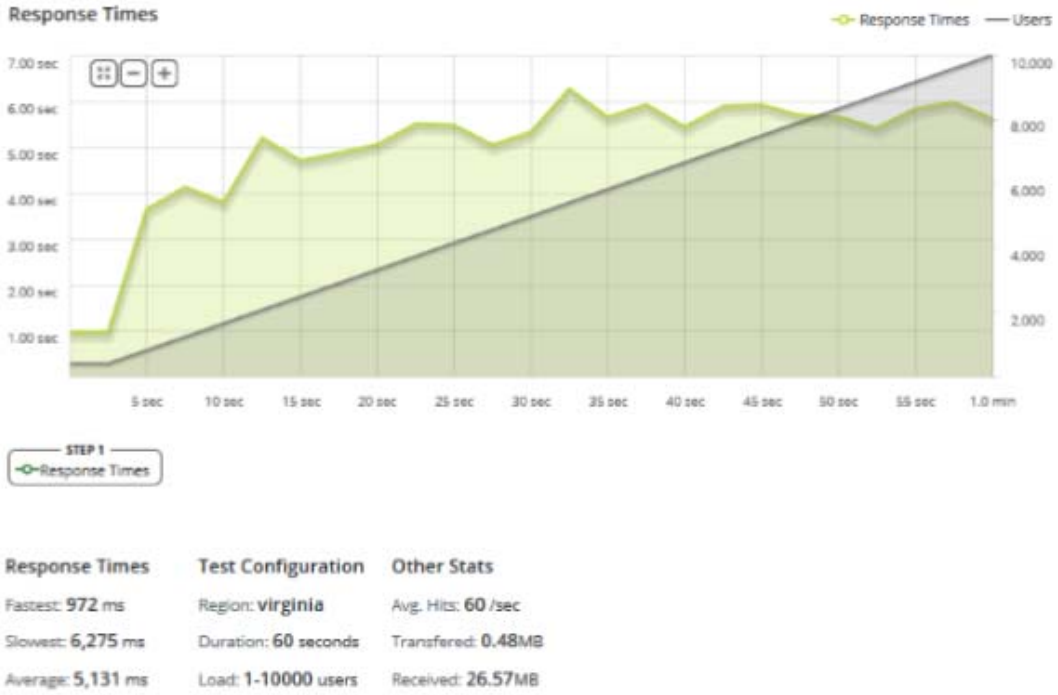

Figure 5: Load Test with 1 to 10000 users

DISCUSSION OF THE RESULTS

After the load tests, all the response times and throughput achieved are higher than the one achieved in performance test after implementation of the performance optimization techniques as shown in Figure 2, this is because there was not a single user on the system during that test. However, they are lower than the one achieved before implementation of the performance optimization technique, this shows that the techniques helped in decreasing the response time and increasing the throughput of the system thereby optimizing its performance even under traffic.

Since the goal of the study is to compare the results of the performance tests before and after performance optimization, we can say that the implementation is very good as the results after the implementation are better than before the implementation.

CONCLUSION

Web-based applications are becoming richer and richer in design and content, at the same time good user experience has become the most sought-after attribute. There is misconception that application desired response time and throughput can be achieved by optimizing server-side (back-end) only. Researches have shown that a greater percentage of page load time is spent on client-side (front-end) and almost half of page load time can be optimized by just focusing on front-end of the application as it is evidenced in this work.

The work shows that implementing the techniques discussed can greatly improve the performance of a web application leading to a good user experience.

REFERENCES

[1] Shailesh Shivakumar, *Presentation-Tier Performance Optimization,* A white paper written for Infosys Limited India, 2017.
[2] *Front End Performance Testing and Optimization.* Available at http://www.agileload.com
[3] Eljona Proko, Ilia Ninka, *Analyzing and Testing Web Application Performance,* Research Inventory: International Journal of Engineering and Science Vol.3, Issue 10 (October 2013), PP 47-50 Issn(e): 2278-4721, Issn(p):2319-6483.
[4] Charu Babber and Neha Bajpai, *Website Performance Analysis based on Component Load testing,* A Review, IJCIT Vol. 1 Issue I.
[5] H. Sarojadevi, *Performance Testing: Methodologies and Tools*, Journal of Information Engineering and Applications, ISSN 22245758 (print) ISSN 2224-896X (online) Vol 1, No.5, 2011.
[6] Radware, *E-Commerce page speed and performance*, State of the Union report, 2014.
[7] Akamai, *E-commerce website performance today, an updated look at consumer reaction to a poor online shopping experience*, Aug 2009.
[8] Quanshu Zhou1, Hairong Ye1, Zuohua Ding, *Performance Analysis of Web Applications Based on User Navigation*, 2012 International Conference on Applied Physics and Industrial Engineering, Physics Procedia 24 (2012) 1319 – 1328.
[9] Ms. S. Sharmila1, Dr. E. Ramadevi, *Analysis of Performance Testing on Web Applications*, International Journal of Advanced Research in Computer and Communication Engineering Vol. 3, Issue 3, March 2014.

[10] Daniel A. Menasce, *Load Testing of Websites*, IEEE 2002.
[11] J. S. Bedi and A. Khanna, *Performance Analysis of Web Applications*, Guru Nanak Dev University, 2013.
[12] M. Elena Gomez, *Performance Analysis of Web Applications,* Group of Discrete Event System Engineering, 2005.
[13] V. Janani and K. Krishnamoorthy, *Evaluation of Cloud-based Performance Testing for online shopping websites,* Indian Journal of Science and Technology, Vol. 8, December 2015.
[14] Aaron Hopkin, *Optimizing Page Load Time*, Source: http://www.die.net accessed on November 20, 2015.
[15] Sheetal S.Patil and Prof. S.D.Joshi, *Identification of Performance Improving Factors for Web Application by Performance Testing*, International Journal of Emerging Technology and Advanced Engineering, ISSN 2250-2459, Volume 2, Issue 8, August 2012.
[16] Kunhua Zhu Junhui Fu Yancui Li, *Research the performance testing and performance improvement strategy in web application*, 2nd International Conference on Education Technology and Computer, 2010, v2 328-332.
[17] J. Yang, U. Sethi, C. Yan, S. Lu, and A. Cheung. *Managing data constraints in database-backed web applications.* In Proceedings of the International Conference on Software Engineering, 2020
[18] Souders, Steve 2007. *High-Performance Web Sites – Essential Knowledge for Frontend Engineers*. O'Reilly, United States.
[19] J. Yang, C. Yan, C. Wan, S. Lu, and A. Cheung. *View-centric performance optimization for database-backed web applications.* In Proceedings of the 41st International Conference on Software Engineering, pages 994–1004. IEEE Press, 2019