

Performance Comparison of dynamic Web scripting Language: A case Study of PHP and ASP.NET

Ogunrinde Mutiat Adebukola, Olorisade Babatunde Kazeem

ABSTRACT - With security, performance, manageability, and scalability requirements for enterprise applications rapidly increasing, choosing the best scripting language in today's competitive environment remains imperative. Scripting language are improving on every front, but enterprises continue to face the arduous but necessary task of evaluating solutions to find the best fit for their unique needs. The paper tends to analyse and compare the performance of the two most popular dynamic scripting languages used in web development, the PHP and ASP.NET. The comparison is based on testing and comparing the performance of the two technologies using software called WAPT. We measure the performance of these two languages using one application benchmarks; an online bookstore. This similarity helps to establish realistic comparison of applications performance. WAPT provides a load and stress testing a tool that provides an easy to use, consistent and cost effective way of testing websites and intranet applications with web interface.

Keywords: Content Generation, Dynamic Web Content, Middleware Technology, Web Applications,

1. INTRODUCTION

Since its introduction, the internet has become pertinent to human day to day endeavour when we use the internet we interact with web Application. Web Applications are used for various purposes such as paying utility bills, social networking, email, online transactions, etc. Web content is increasingly generated dynamically with the use of the dynamic web scripting language, a different approach from the first few years of the Web when virtually all content consisted of static HTML and image files [3][4][12][11][13]. A dynamic web scripting language is a language that responds accordingly to a user's action [14][7][13][11]. They are used to embed interactivity to Web pages that would have been ordinarily static [14][15][10]. They can also automatically supply part information to Web-based forms, among other uses [3][7][11].

Dynamic content is used in many online services that need access to current information such as e-commerce,

electronic banking, etc [4][12][17][10]. Also, it is used to adapt the look-and-feel of Web pages based on the preferences of each user [4][7]. Web Application consists of client (Application) side, Dynamic content (middleware), and a back-end database. Middleware sits between (i.e. in the middle of) the web server and all other backend applications, like databases [18]. It enables a site to move from fixed static pages, which return the same content to every user, to a dynamic application, capable of using programming logic to produce customized content for the user based on their request. All major web sites require middleware. While many scripting languages can function as middleware, choosing the best middleware in today's competitive environment remains imperative.

The performance of a web application is greatly determined by the middleware used [18]. Thus, it is important to choose the best or most suitable middleware for any web application [3][12]. In today's competitive environment, with many scripting languages, making the choice is still difficult, though there has been some study on the performance of scripting languages that there has been a very few once that compare PHP directly with ASP.

In this paper, we evaluate the performance of most popular scripting languages based on performance, fault tolerant, stress and endurance to achieve this; we develop two parallel but similar applications on PHP and the other with ASP. An open tool WAPT is used to measure both applications.

The rest of this paper is organized as follows Section 2 discusses multi-tier web server architecture and the lightweight front-end approach using PHP and ASP.Net. Section 3 covers the related works, section 4 details the methodology used while section 5 discusses the

- Ogunrinde M. A. has MSc in Computer Science from University of Ibadan, Ibadan Nigeria. She is currently working as an Lecturer in Department of Mathematical and Computer Sciences, Fountain University, Osogbo. Osun State of Nigeria. E-mail: bogunrinde@gmail.com
- Olorisade B. K. has MSc in Software Engineering from Universidad Politecnica de Madrid (UPM), Spain and Blekinge Institute of Technology (BTH), Sweden. He currently works as a Lecturer in the Computer Science and Mathematics department of The Nigeria Police Academy, Wudil-Kano State, Nigeria. E-mail: aasimbabatunde@uahoo.co.uk

comparison and the important of the result which is followed by the conclusion in section 6.

2. BACKGROUND

One of the first dynamic scripting languages was Common Gateway Interface (CGI). Using CGI the web programmers were able to produce scripts capable of reading input parameters and return an HTML page meeting specific user requirements [5]. Development of CGI led to the hunt for more sophisticated and robust web scripting languages. Hypertext Preprocessor (PHP) follows the era of CGI. PHP which is an acronym of Hypertext Pre-processor is a server side scripting language, originally designed for producing dynamic pages [8]. It is a widely used general purpose scripting language which can be embedded into Hypertext Transfer Mark-up Language (HTML) source document [14][16]. Afterwards, it can be interpreted by a web server with a PHP processor module generating the web document [18]. It has gained enormous popularity the last few years. It generally runs on a web server taking PHP codes as its input and creating Web pages as its output. It

can also be used for command line scripting and client-side GUI applications. PHP can be deployed on most web servers. It is open source software released under the PHP license. It runs on a variety of operating systems such as Windows, Linux, Mac OS, and most varieties of Unix.

Active Server Pages .NET (ASP.NET) is a web development technology developed by Microsoft. As part of the .NET Framework, ASP.NET allows developers to build dynamic web sites, web applications and web services using compiled languages like VB.NET and C# [12][11]. It follows a code behind model i.e. it deals with the dynamic program code by placing them in a separate file or in a separate tag [9][11]. Using Visual Studio, the development tool from Microsoft, web developers can develop very compelling applications using ASP.NET, with the ease of drag-and-drop server controls. VB.net is perhaps easier to learn, and C# is more used for enterprise applications.

The table below shows a brief comparison between PHP and ASP.NET based on some general parameters.

Table 1. A brief comparison between PHP and ASP.NET

Area	PHP	ASP.NET
Licensing cost	Free licensing	Costly licensing
Platform(s)	Multiple	Only OnWindows
External Hosting	Widely available with zero cost	Widely available although requires fee
Security	Very good	Good
Scalability	Scales very well	Can be difficult to scale
Configuration flexibility	Extremely flexible	Not very flexible
Framework(s)	Many available	Standard framework

In a study to compare the performance of middleware architectures

3. RELATED WORKS

for generating dynamic web content, [4] Evaluates three scripting languages namely PHP, Java servlets, and Enterprise Java Beans (EJB). The study measures the performance of these three languages using two applications benchmarks: an online bookstore that stresses the server back-end and an auction site that stresses the server front-end. It was found out that EJB has lowest performance than both PHP and Java servlets, with Java servlets also having lower performance than PHP.

[2] compares three dynamic web programming technologies from performance in multimedia distribution viewpoint. In particular, the paper examines Sun Microsystems's Java Server Pages (JSP), Microsoft's Active Server Pages (ASP) and the more recent ASP.NET. The comparison is based on testing applications that distribute multiple images from an Oracle 9i database to Web-enabled clients. The comparison is accomplished by developing a prototype application, testing it, and eliciting the foremost factors contributing to the final performance and suitability of the different technologies. The results shows that ASP and JSP significantly outperforms the ASP.NET while no significant different is noticed between the two. This result was similar to the study in [6] except that the JSP was replaced by PHP.

PHP, Java servlets and CGI was compared as approaches for web-to-database applications by [1] Their benchmark test is restricted to data retrieval (read) operations, they only use a configuration where the Java servlet server runs on the Web Server and even with this configuration, servlets outperform the two scripting languages. However, they use PHP3. Their result shows that Java servlets have a slightly larger overhead than PHP, but the flexibility of servlets allows architectures where the load is balanced among several servers.

[11] carried out sophisticated experiments to ascertain the performances of the three web scripting languages viz. PHP, JSP and ASP.NET. Their experiment is based on the self-written codes and built-in modules. They concluded from the graph in their report that no language excels under all circumstances. Considering numerical calculation, they concluded that PHP works best out of the three languages under consideration as shown by the action of the three languages on factorial benchmark. It holds for both self-written codes and built-in modules. On the contrary, considering comparison based problems such as merge sort, JSP comes out to be the clear winner out of the

three languages for both self-written and built-in modules. Thus, they inferred that in case of complex problems such as Dijkstra's algorithm and problem pertaining to string manipulations such as palindrome checking that ASP.NET performs stupendously.

[15] reported that when implementing a web server system which will never experience high load, or in which performance, throughput, and reliability under high load is not an issue, then the use of any of the analyzed languages or web servers will achieve similar performance results. If outstanding performance and throughput is the primary goal, then the use of JSP over PHP is advisable. However, if a 5-10% difference in throughput and performance is acceptable, then the implementer of a web system can achieve similar results using either PHP or JSP. In which case, other requirements such as developer language familiarity and programming efficiency, maintainability, security, reliability, middleware compatibility, etc. would be the deciding factors. It is also reassuring to developers of both language runtimes and web servers, that enhancements to either can offer performance improvements to the community.

4. METHODOLOGY

This paper compares the performance of dynamic web scripting languages used for web application development. The work is divided into 2 phases. In Phase I, two identical web applications was developed i.e. ecommerce website (applications to purchase books online) in PHP and ASP.NET. In phase II, an automated test was conducted to determine and analyze the performance of Applications developed both in PHP and ASP.NET. This similarity helps to establish realistic comparison of applications performance. The applications' performance is measured with help of software called WAPT. WAPT provides an environment for load, stress and performance testing of web sites and applications with web interface. It is used primarily for testing websites, web servers and intranet applications. It is used to test and analyze application

performance under different phases of development such as the design stage and the coding stage, but in this case it is used to test and analyse the system performance between two web technologies i.e. ASP.NET and PHP.

5. COMPARISON AND RESULT

Response time which refers to the time interval for a client to send and receive request is selected as key criteria for

comparison in terms of speed for any of the two applications developed using two different technologies. i.e. ASP.NET and PHP. These two technologies were kept the same during the application development. Following subsections presents the detailed result of the tests carried out.

5.1 Performance Test

Performance test is used to measure the performance characteristics of a web site. Performance is usually measured as average, minimum and maximum response times. Average response time is the most important indicator of performance.

Table 1: Performance test result for PHP

Profile	Successful sessions	Failed sessions	Successful pages	Failed pages	Successful hits	Failed hits	Total KBytes sent	Total KBytes received	Avg Response time, sec (with page elements)
performancephp	2,715	0	2,738	0	2,738	0	669	10,045	0.03 (0.03)
Total	2,715	0	2,738	0	2,738	0	669	10,045	0.03 (0.03)

Response time, sec (with page elements)

Name	Time	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
performancephp		290	282	244	247	249	257	255	278	270	343	2,715
	Min	0 (0)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0 (0)
	Avg	0.02 (0.02)	0.02 (0.02)	0.02 (0.02)	0.02 (0.02)	0.02 (0.02)	0.04 (0.04)	0.04 (0.04)	0.04 (0.04)	0.04 (0.04)	0.04 (0.04)	0.03 (0.03)
	Avg90	0.03 (0.03)	0.03 (0.03)	0.05 (0.05)	0.05 (0.05)	0.04 (0.04)	0.08 (0.08)	0.09 (0.09)	0.05 (0.05)	0.07 (0.07)	0.07 (0.07)	0.06 (0.06)
	Max	0.12 (0.12)	0.13 (0.13)	0.3 (0.3)	0.25 (0.25)	0.13 (0.13)	0.33 (0.33)	0.39 (0.39)	0.08 (0.08)	0.17 (0.17)	0.27 (0.27)	0.39 (0.39)

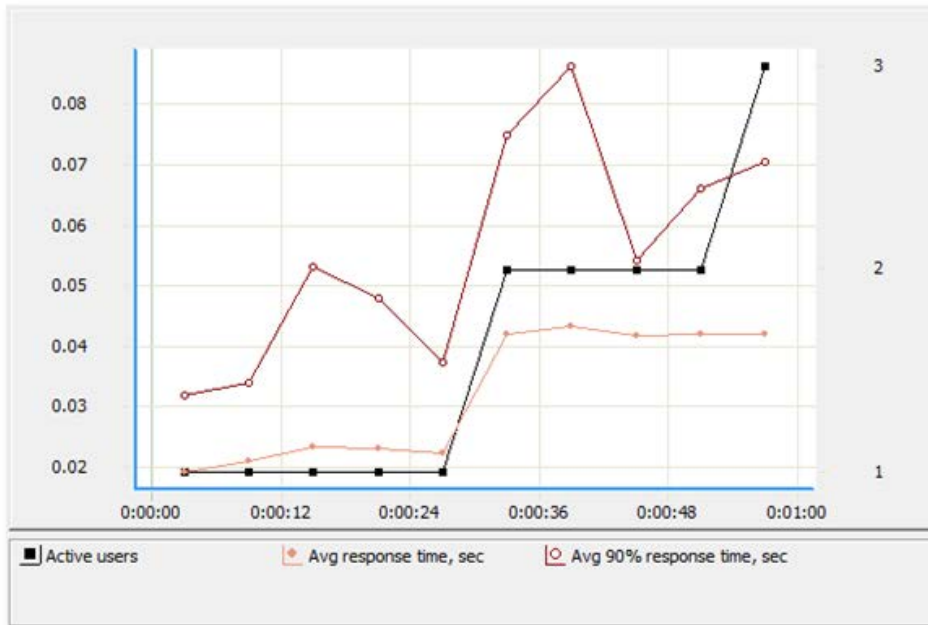


Figure 1: Performance graph result for PHP

The table 1 and figure 1 above shows the performance test result of 20 users and all the sessions (2,715) and pages (2,738) was successful. It reveals that 669 Kbytes was sent to the server and received 10, 045Kbytes during the test run. The average response time without page elements indicate that it

takes 0.03s for each user to start reading a page and also the average response time with page elements for each user to download graphics and flash images is 0.03s. The graph also indicates that the response time from the server does not increase as the number of user increases.

Table 2 Performance test result for ASP.NET

Profile	Successful sessions	Failed sessions	Successful pages	Failed pages	Successful hits	Failed hits	Total KBytes sent	Total KBytes received	Avg Response time, sec (with page elements)
asp.netperformancetesting	1,323	0	1,366	0	1,366	0	355	37,486	0.06 (0.06)
Total	1,323	0	1,366	0	1,366	0	355	37,486	0.06 (0.06)

Response time, sec (with page elements)

Name	Time	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
asp.netperformancetesting		140	142	102	111	130	134	128	134	120	182	1,323
	Min	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.08 (0.08)	0.08 (0.08)	0.03 (0.03)	0.03 (0.03)
	Avg	0.04 (0.04)	0.04 (0.04)	0.04 (0.04)	0.04 (0.04)	0.04 (0.04)	0.04 (0.04)	0.09 (0.09)	0.09 (0.09)	0.1 (0.1)	0.07 (0.07)	0.06 (0.06)
	Avg90	0.06 (0.06)	0.05 (0.05)	0.05 (0.05)	0.05 (0.05)	0.07 (0.07)	0.06 (0.06)	0.19 (0.19)	0.11 (0.11)	0.13 (0.13)	0.24 (0.24)	0.14 (0.14)
	Max	0.13 (0.13)	0.08 (0.08)	0.09 (0.09)	0.06 (0.06)	0.28 (0.28)	0.13 (0.13)	0.91 (0.91)	0.14 (0.14)	0.19 (0.19)	1.22 (1.22)	1.22 (1.22)

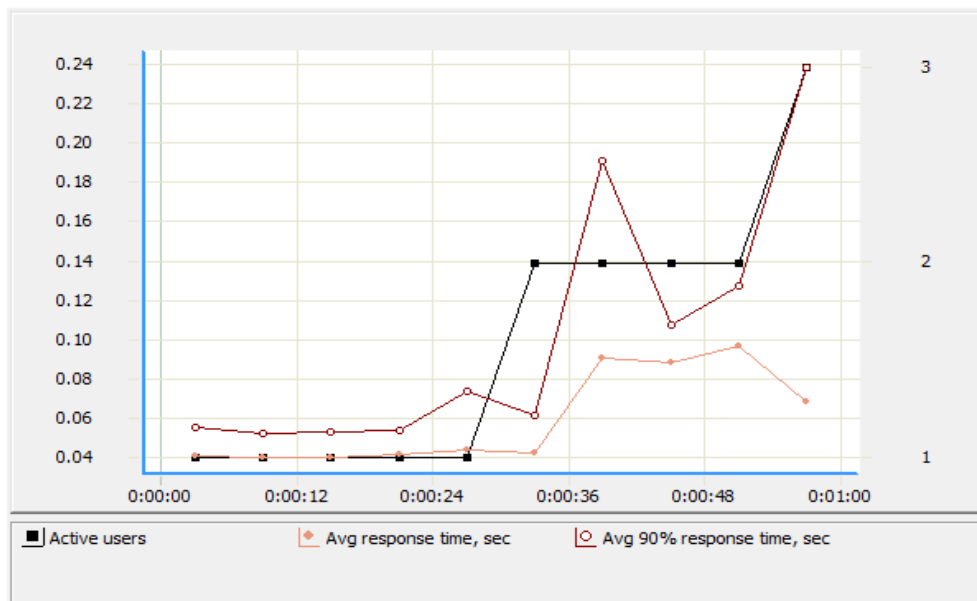


Figure 2: Performance graph test result for ASP.NET

Table 2 and figure 2 shows the performance test result of 20 users and all the sessions (1,323) and pages (1,366) was successful. It reveals that 355 Kbytes was sent to the server and received 37,486Kbytes during the test run. The average response time without page elements indicate that it takes 0.06s for each user to start reading a page and also the average response time with page elements for each user to download graphics and flash images is 0.06s.

The performance test of both applications shows that the total Kbyte sent to the server during PHP test is higher than ASP.NET and also the total Kbytes received in ASP.NET is higher than PHP. This bring about the increase in ASP.NET response time. Results show that PHP has better performance than ASP.NET respectively.

5.2 Stress Test

Stress tests steadily increase the load on the system beyond the maximum design load. This type of testing has a dual purpose:

It tests the failure behaviour of the system. In real life, an unexpected combination of events may occur. It may result in much increase of load on the system. This load may exceed the maximum expected load. Stress testing determines if overloading the system results in loss of data or user service. This kind of test helps finding defects and errors that don't reveal themselves in usual situation.

Table 3: stress test result for PHP

Profile	Successful sessions	Failed sessions	Successful pages	Failed pages	Successful hits	Failed hits	Total KBytes sent	Total KBytes received	Avg Response time, sec (with page elements)
stresstestphp	2,698	0	2,720	0	2,720	0	665	9,972	0.03 (0.03)
Total	2,698	0	2,720	0	2,720	0	665	9,972	0.03 (0.03)

Response time, sec (with page elements)

Name	Time	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
stresstestphp		286	283	200	302	294	272	225	231	272	333	2,698
	Min	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)
	Avg	0.02 (0.02)	0.02 (0.02)	0.03 (0.03)	0.02 (0.02)	0.02 (0.02)	0.04 (0.04)	0.05 (0.05)	0.05 (0.05)	0.04 (0.04)	0.04 (0.04)	0.03 (0.03)
	Avg90	0.03 (0.03)	0.03 (0.03)	0.19 (0.19)	0.03 (0.03)	0.03 (0.03)	0.07 (0.07)	0.15 (0.15)	0.16 (0.16)	0.07 (0.07)	0.08 (0.08)	0.1 (0.1)
	Max	0.06 (0.06)	0.08 (0.08)	1.84 (1.84)	0.05 (0.05)	0.09 (0.09)	0.3 (0.3)	1.09 (1.09)	1.14 (1.14)	0.27 (0.27)	0.48 (0.48)	1.84 (1.84)



Figure 3: stress test graph result for PHP

Table 4 : Stress test result for ASP.NET

Profile	Successful sessions	Failed sessions	Successful pages	Failed pages	Successful hits	Failed hits	Total KBytes sent	Total KBytes received	Avg Response time, sec (with page elements)
stresstestaspnet	1,238	0	1,250	0	1,250	0	325	34,303	0.06 (0.06)
Total	1,238	0	1,250	0	1,250	0	325	34,303	0.06 (0.06)

Response time, sec (with page elements)

Name	Time	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
stresstestaspnet		136	107	117	108	135	136	114	102	123	160	1,238
	Min	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)
	Avg	0.04 (0.04)	0.05 (0.05)	0.05 (0.05)	0.05 (0.05)	0.04 (0.04)	0.05 (0.05)	0.1 (0.1)	0.08 (0.08)	0.07 (0.07)	0.06 (0.06)	0.06 (0.06)
	Avg90	0.05 (0.05)	0.1 (0.1)	0.08 (0.08)	0.09 (0.09)	0.06 (0.06)	0.07 (0.07)	0.69 (0.69)	0.26 (0.26)	0.23 (0.23)	0.11 (0.11)	0.25 (0.25)
	Max	0.08 (0.08)	0.27 (0.27)	0.19 (0.19)	0.2 (0.2)	0.13 (0.13)	0.09 (0.09)	4.95 (4.95)	1.44 (1.44)	0.61 (0.61)	0.2 (0.2)	4.95 (4.95)

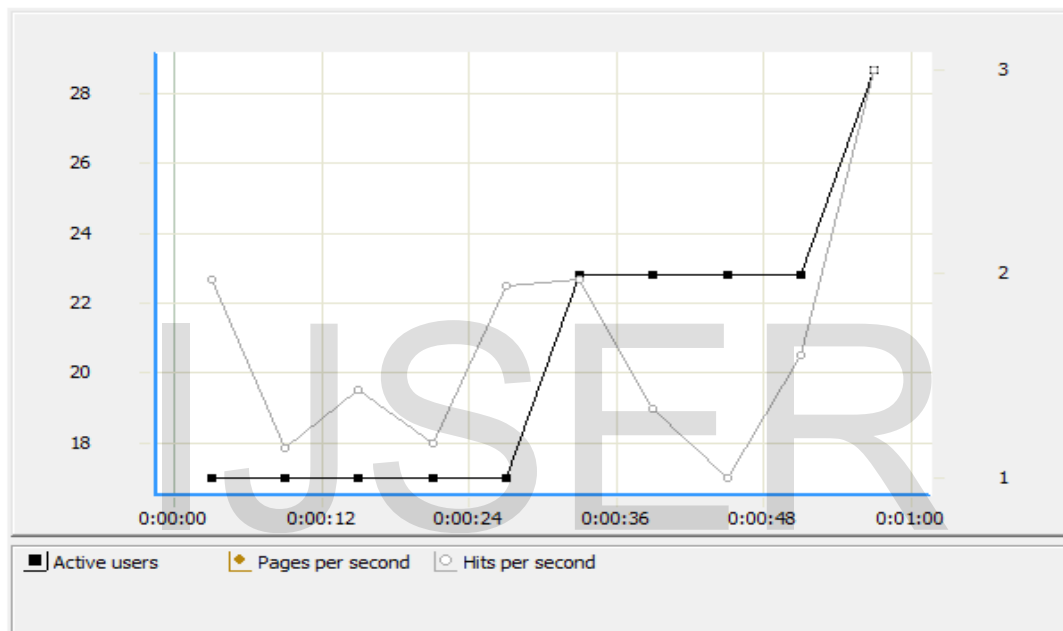


Figure 4: Stress test graph result for ASP.NET

Table 3 and 4 shows the average response time for ASP.NET clients is 0.06s and for PHP clients is 0.03s. Maximum response for ASP.NET clients is 0.25s and for PHP clients is 1.84s. The maximum response time for ASP.NET is better compared to PHP. And this indicates a slightly bad response time from PHP. Considering the average response time for both applications, the results shows that PHP performs better than ASP.NET when load is steadily increase on both applications.

5.3 Endurance Test

This type of test checks the ability of a web site to withstand the specified load for a long period of time. In such conditions the performance degradation can be a result of memory leaks or other server resources allocation problem. The endurance testing is usually performed for several hours. The goal is to confirm that the web site performance during different test phase does not differ significantly. The result in table 5 and table 6 shows that PHP can withstand a specified load for a long period of time than ASP.NET.

Table 5: Endurance test result for PHP

Profile	Successful sessions	Failed sessions	Successful pages	Failed pages	Successful hits	Failed hits	Total KBytes sent	Total KBytes received	Avg Response time, sec (with page elements)
endurancetestinphp	110,872	0	110,967	0	110,967	0	27,101	406,808	0.55 (0.55)
Total	110,872	0	110,967	0	110,967	0	27,101	406,808	0.55 (0.55)

Response time, sec (with page elements)

Name	Time	0:00:00-0:06:00	0:06:00-0:12:00	0:12:00-0:18:00	0:18:00-0:24:00	0:24:00-0:30:00	0:30:00-0:36:00	0:36:00-0:42:00	0:42:00-0:48:00	0:48:00-0:54:00	0:54:00-1:00:00	Total
endurancetestinphp		15,622	11,686	11,831	11,368	11,131	10,248	8,696	8,250	11,380	10,660	110,872
	Min	0.01 (0.01)	0.08 (0.08)	0.09 (0.09)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)	0.01 (0.01)
	Avg	0.35 (0.35)	0.56 (0.56)	0.52 (0.52)	0.5 (0.5)	0.59 (0.59)	0.53 (0.53)	0.73 (0.73)	0.72 (0.72)	0.51 (0.51)	0.65 (0.65)	0.55 (0.55)
	Avg90	0.94 (0.94)	1.72 (1.72)	1.73 (1.73)	1.83 (1.83)	1.88 (1.88)	1.83 (1.83)	2.35 (2.35)	2.46 (2.46)	1.64 (1.64)	2.03 (2.03)	1.82 (1.82)
	Max	8.95 (8.95)	9.58 (9.58)	13.7 (13.7)	14.9 (14.9)	30.9 (30.9)	21.3 (21.3)	23.0 (23.0)	25.9 (25.9)	23.0 (23.0)	23.6 (23.6)	30.9 (30.9)



Figure 5 Endurance test graph result for PHP

Table 6: Endurance test result for ASP.NET

Profile	Successful sessions	Failed sessions	Successful pages	Failed pages	Successful hits	Failed hits	Total KBytes sent	Total KBytes received	Avg Response time, sec (with page elements)
endurancetestaspnet	80,548	0	80,593	0	80,593	0	20,946	2,211,801	0.6 (0.6)
Total	80,548	0	80,593	0	80,593	0	20,946	2,211,801	0.6 (0.6)

Response time, sec (with page elements)

Name	Time	0:00:00-0:06:00	0:06:00-0:12:00	0:12:00-0:18:00	0:18:00-0:24:00	0:24:00-0:30:00	0:30:00-0:36:00	0:36:00-0:42:00	0:42:00-0:48:00	0:48:00-0:54:00	0:54:00-1:00:00	Total
endurancetestaspnet		7,926	8,167	8,418	8,136	8,584	8,054	7,806	7,740	8,093	7,624	80,548
	Min	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.03 (0.03)	0.25 (0.25)	0.19 (0.19)	0.03 (0.03)
	Avg	0.5 (0.5)	0.58 (0.58)	0.56 (0.56)	0.51 (0.51)	0.6 (0.6)	0.5 (0.5)	0.62 (0.62)	0.67 (0.67)	0.66 (0.66)	0.83 (0.83)	0.6 (0.6)
	Avg90	1.9 (1.9)	1.88 (1.88)	1.97 (1.97)	2.67 (2.67)	1.95 (1.95)	1.71 (1.71)	2.53 (2.53)	1.84 (1.84)	1.38 (1.38)	1.67 (1.67)	2.02 (2.02)
	Max	24.7 (24.7)	14.9 (14.9)	20.9 (20.9)	59.0 (59.0)	9.83 (9.83)	6.16 (6.16)	40.2 (40.2)	12.9 (12.9)	6.35 (6.35)	9.69 (9.69)	59.0 (59.0)

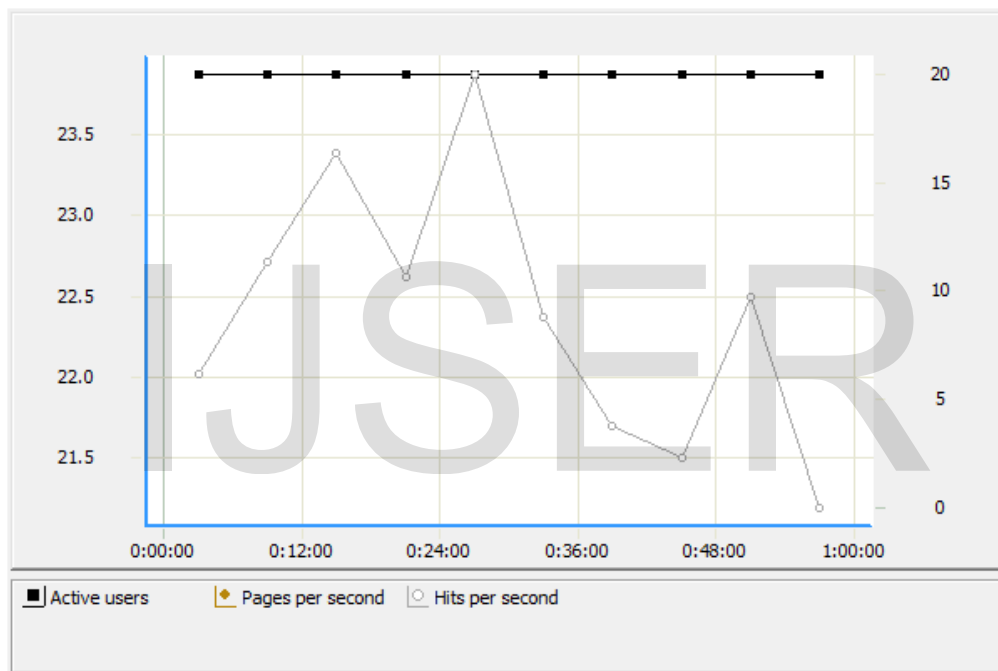


Figure 6 Endurance test graph result for ASP.NET

5.4 Failure Test

The failure test revealed the following for both ASP.NET and

PHP:

Failed Session: 0

Network Error: 0

Failed Page: 0

Failed Hit: 0

Response Session: All pages and hits, with zero error, Http Error: 0

Table 7: Failure test for PHP

Response codes											
Code	Request									Pages	Hits
performancephp	-									2,715	2,715
200 OK	performancephp.All									2,715	2,715
200 OK	performancephp.page_1: http://localhost:8080/project/									2,715	2,715

HTTP errors, %											
Profile	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
performancephp	0	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0	0

Network errors, %											
Profile	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
performancephp	0	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0	0

Timeouts, %											
Profile	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
performancephp	0	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0	0

Total errors, %											
Profile	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
performancephp	0	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0	0

Table 8: Failure test for ASP.NET

Response codes											
Code	Request									Pages	Hits
asp.netperformancetesting	-									1,323	1,323
200 OK	asp.netperformancetesting.All									1,323	1,323
200 OK	asp.netperformancetesting.page_1: http://localhost/peju_project/store/index.aspx									1,323	1,323

HTTP errors, %											
Profile	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
asp.netperformancetesting	0	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0	0

Network errors, %											
Profile	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
asp.netperformancetesting	0	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0	0

Timeouts, %											
Profile	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
asp.netperformancetesting	0	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0	0

Total errors, %											
Profile	0:00:00-0:00:06	0:00:06-0:00:12	0:00:12-0:00:18	0:00:18-0:00:24	0:00:24-0:00:30	0:00:30-0:00:36	0:00:36-0:00:42	0:00:42-0:00:48	0:00:48-0:00:54	0:00:54-0:01:00	Total
asp.netperformancetesting	0	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0	0

6. CONCLUSION

This paper tends to analyze and compare the performance of ASP.NET and PHP 5.3.0 frameworks. Two web applications was developed i.e. online bookstore using these frameworks.

A number of automated tests were performed to measure the performance of these applications such as performance test, stress test and endurance tests. Both applications give an ideal result. The result shows that the requests take minimum time

to perform any action on both frameworks. But since the response time of PHP is lower than ASP.NET and PHP has a good performance. However, in stress test PHP also performed better than ASP.NET with the response time of 0.03s and 0.06s respectively. In endurance testing PHP also performed better than ASP.NET. In conclusion, the test result shows that PHP works best out of the two scripting languages.

7. REFERENCES

- [1] Amanda, W., W. Haibo, et al. (2000). Performance Comparison of Alternative Solutions

For Web-To-Database Applications. Southern Conference on Computing.

[2] Amza, C. (2002). Specification and implementation of dynamic Web site benchmarks. 5th IEEE Workshop on Workload Characterization, IEEE.

[3] Apte, V., T. Hansen, et al. (2003). "Performance comparison of dynamic web platforms." Computer Communications **26**(8): 888-898.

[4] Cecchet, E., A. Chanda, et al. (2003). Performance comparison of middleware architectures for generating dynamic web content. Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware, Springer-Verlag New York, Inc.

[5] Coar, K. and D. Robinson (1999). The WWW Common Gateway Interface. Internet draft, version 1.1.

[6] Hartman, H. (2001). "Tools for dynamic Web sites: ASP vs PHP vs ASP .NET." Seybold Report Analysing Publishing Technologies **15339211**(12): 1-12.

[7] Karve, A., T. Kimbrel, et al. (2006). Dynamic placement for clustered web applications. Proceedings of the 15th international conference on World Wide Web, ACM.

[8] Ledorf, R. (2002). Programming PHP. California, O'Reilly Media.

[9] MacDonald, M. (2002). ASP.NET: The Complete Reference. New York, McGraw-Hill Companies.

[10] Peña-Ortiz, R., J. A. Gil, et al. (2013). "Analyzing web server performance under dynamic user workloads." Computer Communications **36**(4): 386-395.

[11] Ranjan, A., R. Kumar, et al. (2012). "A comparative study between dynamic web scripting languages." Data Engineering and Management: 288-295.

[12] Swales, D., D. Sewry, et al. (2003). A Performance Comparison of Web Development Technologies to Distribute Multimedia across an Intranet. Proceedings of Southern African Telecommunication Networks and Applications Conference (SATNAC).

[13] Swaminathan, S., P. Guillaume, et al. (2007). "Analysis of Caching and Replication Strategies for Web Applications." IEEE Internet Computing.

[14] Titchkosky, L., M. Arlitt, et al. (2003). "A performance comparison of dynamic web

technologies." ACM SIGMETRICS Performance Evaluation Review **31**(3): 2-11.

[15] Trent, S., M. Tatsubori, et al. (2008). "Performance comparison of PHP and JSP as server-side scripting languages." Middleware 2008: 164-182.

[16] Vanhatupa, J. M. (2011). On the development of browser games-Technologies of an emerging genre. Next Generation Web Services Practices (NWeSP), 2011 7th International Conference on, IEEE.

[17] Zeng, L., B. Benatallah, et al. (2004). "QoS-aware middleware for web services composition." Software Engineering, IEEE Transactions on **30**(5): 311-327.

[18] Zwanzinger, T. (2004). Testing and Improving Web Application Performance.