Performance Testing Mentoring Program 2022-2023

Task 12 Part 3

Created by: Serhii Diedov

Date: May 2023

BlogEngine.NET 3.2
PERFORMANCE TESTING REPORT

1 Goal

The purpose of testing determined by the customer is to determine with which data source (file system or DB) the server side of the application BlogEngine.NET 3.2 (web) is more productive and stable, including with an increase in data volume, and also has the best potential for scaling.

For this purpose, the following group of tests will be carried out:

1 Perform capacity testing (for 1000 text posts) and compare results with both file system and DB data source.

2 Perform scalability testing (for 1000 text posts) and compare results with both file system and DB data source using the following approach:

2.1 Approach to determine the impact of the number of CPUs on system performance:

a) Set the size of the memory as bigger as possible for the virtual machine (6Gb).

b) CPU scaling and perform load testing under regular load (~70% of the system capacity) for different numbers of CPUs: 1, 2, 3, 4, 6.

2.2 Approach to determine the impact of the RAM on system performance:

a) Set the number of CPUs as big as possible (6).

b) Scaling RAM and perform load testing under regular load (~70% of the system capacity) for different sizes of RAM: 2Gb, 3Gb, 4Gb (if possible), 6Gb (if possible).

3 Perform regular load testing (~70% of the system capacity) with the parameters based on the results of capacity testing (for 1000 text posts) **and perform stress testing** with a load of 120%, 200%, 300% and 500% system capacity. The goal is to determine the ability of the system to cope with the stress load and to identify the consequences of such a load on the system.

4 Perform volume testing and compare results with both file system and DB data source using the following approach:

a) Testing under regular load (~70% of the system capacity) with different number of the text posts 100, 1000, 2000, 5000.

b) Testing under regular load (~70% of the system capacity)with media information and compare results for 1000 text posts and 1000 posts with a text and attached 1Mb photo

5 To ensure system stability under long-term load, **additionally perform long-time testing** under low load (~30% of the system capacity)

High-level description of the load model: Testing for a combined scenario describing scenarios for simultaneous work with the BlogEngine.NET 3.2. by users with the roles admin, editor, anonymous.

Testing scenario: simulating user behavior on the site, where:
Admin - Open home page, log in as Admin user, open Admin's page, check number of Anonymous users and decide to add or delete user, finally Log out

Editor - Open home page, log in as an Editor user, open a predefined date, open a random post and edit it, rapid it 50 times, and finally Log out.

Anonymous user - visits different pages, opens posts and leave comments.

Test environment: test application and load generator are running on the same computer. Application

is executed on the virtual machine and the load generator is executed on the host.

---

2 Main results from test runs

Capacity testing results:

After switching data source from file system to DB Capacity, the system point became less explicit and decreased slightly, amounting to 31 concurrent users with 33.4 successful transactions per second (compared to 38 concurrent users with 36.2 tr/sec for the file system).

Scalability testing results:

Scalability testing showed that when using the database as a data source application BlogEngine .NET 3.2 (web) does NOT scale with different amounts of RAM (2048 MB - 6144 MB) while keeping the maximum value of CPUs (6) unchanged.

Also when using the database as a data source application BlogEngine .NET 3.2 (web) does NOT noticeably scale with different amounts of CPUs (1,2,4,6) while maintaining the same maximum RAM value (6144 MB).

Important: It is important to say that IIS (Internet Information Services) doesn't handle with load test with more than one CPU when dealing with the file system as a data source. IIS stops working after some time (3-5 minutes after the start of the test). This behavior looks critical for an application that uses the file system as a data source.

Volume testing results:

After switching data source from file system to DB volume testing application BlogEngine .NET 3.2 (web) with test data as different numbers of text posts (100, 1000, 2000) and for 1000 posts with a text and attached 1Mb photo showed majority independence key performance indicators on the number of posts and their types existing in the system.

At the same time, both for the system whose data source is the file system and the database, the indicators for 5000 text posts are significantly different. So the average Throughput for a system whose data source is a file system is reduced by 3 times compared to 100 text posts, but for a system whose data source is a database, the influence is much less, only by one third.

Results of long-time testing:

After switching data source from file system to DB volume testing application BlogEngine .NET 3.2 (web) it was additionally perform long-time testing under low load (~30% of the system capacity) for 8 hours. The main purpose of this test was to identify possible errors associated with memory leaks and other unexpected behavior associated with the duration and slow growth of data.

The application BlogEngine .NET 3.2 (web) database-sourced showed higher mean Throughput stability throughout the test, lower mean CPU usage and mean error rate, and lower deviations in mean Memory usage.

Results of regular load testing and stress testing:

For clarity, the results of stress testing with a load of 120%, 200%, 300% and 500% system capacity are shown together with the results of regular load testing (70% of the system capacity).

Stress testing was performed for application BlogEngine .NET 3.2 (web) when working with the database as a data source. As a result of stress testing, the system showed a fairly high stability. As the load increased, the average response time was expected to increase, which for the most time-consuming requests could increase ten or more times.

Error rate: The error rate generally stays under 1%, with the exception of the "Editpost" (TC_Ed_Editpost) request, which is between 10% and 53% of this type of request. The error is reproduced through the GUI while there is a load. This percentage of errors looks unacceptable in terms of user experience for the "editor" role.

3 Summaries on results

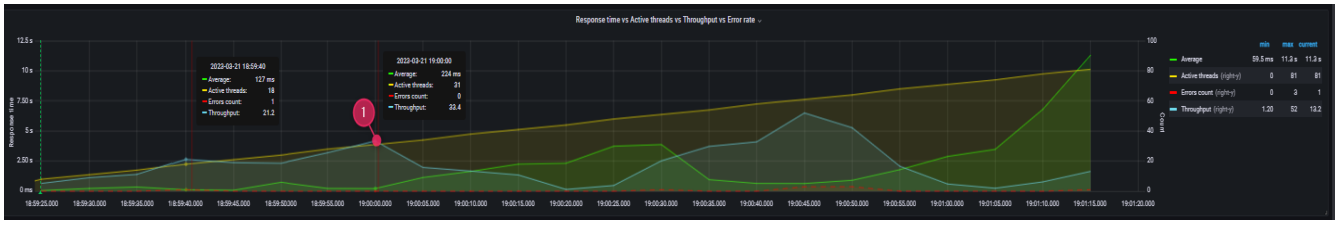
Capacity testing results:

After switching data source from file system to DB Capacity, the system point became less explicit and decreased slightly, amounting to 31 concurrent users with 33.4 successful transactions per second (compared to 38 concurrent users with 36.2 tr/sec for the file system).

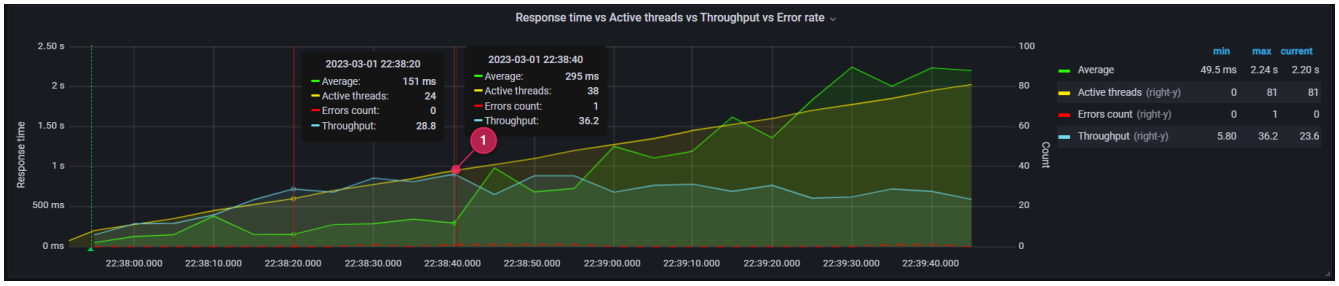
Comfort zone [response time is not growing, and throughput is stable growing] – up to 18 concurrent users and up to 21.2 successful transactions per second (24 concurrent users and up to 28.8 tr/sec for the file system).

Performance degradation zone [response time is lightly growing, and throughput is growing more slowly than in the comfort zone] - in the range between 18 and 31 concurrent users, and throughput from 21.2 to 33.4 successful transactions per second (in the range between 24 and 38 concurrent users, and throughput from 28.8 to 36.2 tr/sec for the file system).

Response time vs Active threads vs Throughput vs Error rate for the database as a data source.



Response time vs Active threads vs Throughput vs Error rate for the file system as data source.



Scalability testing results:

Scalability testing showed that when using the database as a data source application BlogEngine .NET 3.2 (web) does NOT scale with different amounts of RAM (2048 MB - 6144 MB) while keeping the maximum value of CPUs (6) unchanged. **Similar behavior has been noted for the file system as a data source.**

As can be seen from the table, increasing RAM reduces the average Memory usage and does not significantly affect other key performance indicators.

Performance Indicators	6x2Gb	DB_6x2Gb	6x3Gb	DB_6x3Gb	6x4Gb	DB_6x4Gb	6x6Gb	DB_6x6Gb
Throughput max (rps)	46	38	44	38	46	38	44	36
Throughput avg (rps)	**38**	**32**	**37**	**32**	**40**	**32**	**37**	**30**
CPU usage max (%)	48	53	50	52	83	31	88	49
CPU usage avg (%)	**30**	**26**	**33**	**28**	**36**	**17**	**30**	**33**
Memory usage max (%)	90	92	81	83	75	77	56	61
Memory usage avg (%)	**89**	**87**	**77**	**76**	**68**	**67**	**51**	**52**
IIS stopped during the test	Y	N	Y	N	N	N	Y	N
An effective duration of test (sec)	**222**	903	**331**	905	**906**	904	**178**	909
Transactions count	8264	28828	12233	28877	35976	29062	6398	27160
Error count	28	150	78	97	195	128	40	182
Averge error rate (%)	**0.3**	**0.5**	**0.6**	**0.3**	**0.5**	**0.4**	**0.6**	**0.7**

At the same time, tests have shown that application **BlogEngine .NET 3.2 (web) does noticeably scale with different amounts of CPUs (1,2,4,6)** while maintaining the same maximum RAM value (6144 MB) while **working with the file system but it does NOT scale when working with the database as a data source.**

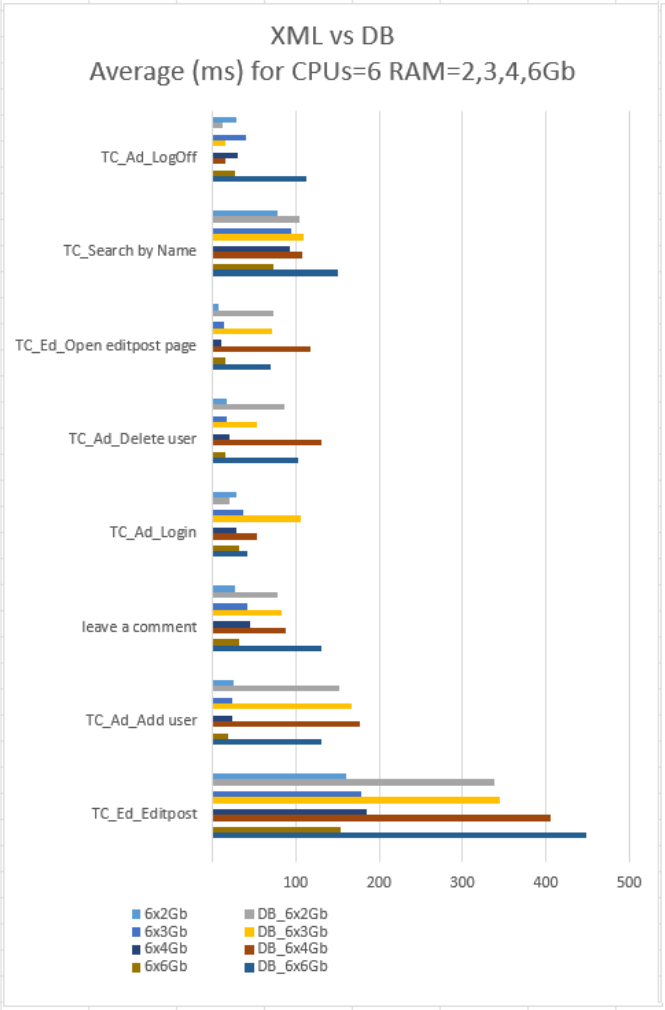
As can be seen from the table, when working with a database as a data source, an increase in CPUs reduces the average CPU usage and does not significantly affect other key performance indicators.

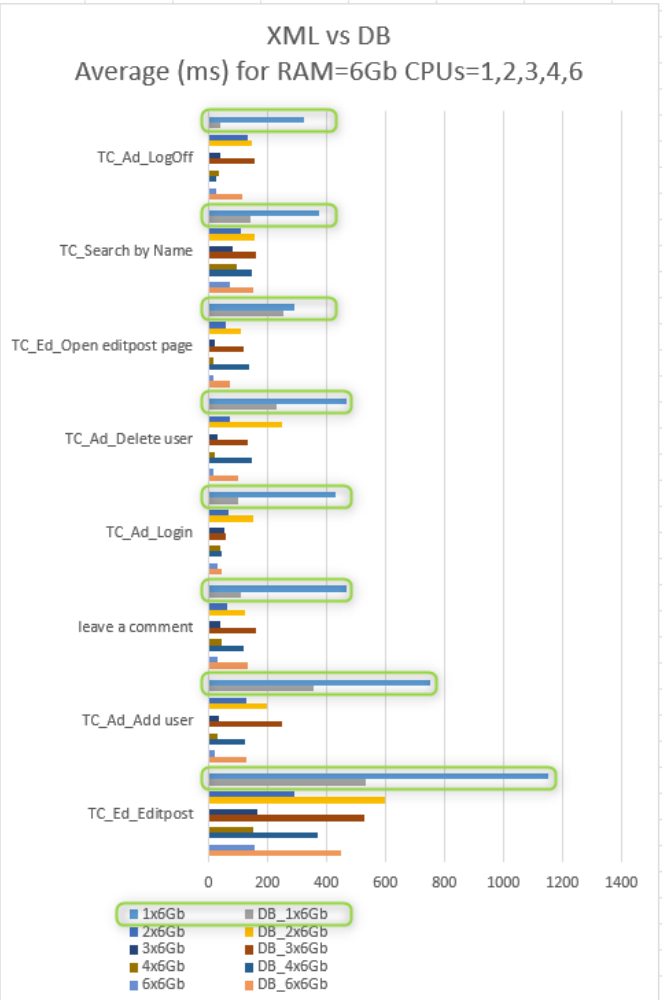
Performance Indicators	1x6Gb	DB_1x6Gb	2x6Gb	DB_2x6Gb	3x6Gb	DB_3x6Gb	4x6Gb	DB_4x6Gb	6x6Gb	DB_6x6Gb
Throughput max (rps)	42	37	47	35	44	35	46	38	44	36
Throughput avg (rps)	**29**	**31**	**39**	**30**	**38**	**30**	**39**	**31**	**37**	**30**
CPU usage max (%)	100	100	100	96	72	100	79	89	88	49
CPU usage avg (%)	**96**	**79**	**78**	**76**	**52**	**61**	**48**	**46**	**30**	**33**
Memory usage max (%)	55	55	52	61	58	55	57	62	56	61
Memory usage avg (%)	**51**	**53**	**50**	**54**	**53**	**50**	**53**	**52**	**51**	**52**
IIS stopped during the test	N	N	Y	N	Y	N	Y	N	Y	N
An effective duration of te	909	905	**589**	907	**249**	909	**319**	908	**178**	909
Transactions count	19970	27611	22922	27503	9260	27448	12227	27965	6398	27160
Error count	86	81	78	175	34	167	56	154	40	182
Averge error rate (%)	**0.4**	**0.3**	**0.3**	**0.6**	**0.4**	**0.6**	**0.5**	**0.6**	**0.6**	**0.7**

Note: It is important to say that **IIS (Internet Information Services) doesn't handle with load test with more than one CPU when dealing with the file system as a data source**. IIS stops working after some time (3-5 minutes after the start of the test). The existing level of logging did not allow to identify the error that leads to this behavior. The identified errors in some cases did not lead to IIS stopping, in other cases, IIS stopped 30-40 seconds after they appeared. Our comparison based on working time when IIS continued to respond.

Given the above, although we see that the response time for the most time-consuming requests for the file system as a data source for most configurations is less, but the system was extremely unstable under these configurations. At the same time, for any of the configurations when working with the database as a data source, the system remained stable.

So if we compare the stable configuration with both file system and DB data source (marked with green frames on the graph), then we can see that the average response time for DB as a data source is 2 or more times faster compared to the file system as a data source. So for DB as a data source, the average Throughput of 31rps (29rps for the file system) is also significantly lower than the average CPU usage which is 79% (93% for the file system).





Error rate: The error rate generally stays under 1%, with the exception of the "Editpost" (TC_Ed_Editpost) request, which is between 10% and 53% of this type of request. The error is reproduced through the GUI while there is a load. This percentage of errors looks unacceptable in terms of user experience for the "editor" role.

	6x2Gb	DB_6x2Gb	6x3Gb	DB_6x3Gb	6x4Gb	DB_6x4Gb	6x6Gb	DB_6x6Gb
Error rate TC_Ed_Editpost	27%	30%	52%	30%	46%	28%	53%	46%
Error rate leave a comment	1%	1%	1%	0%	1%	1%	2%	1%
Error rate Open Random post	-	0%	0%	-	0%	0%	0%	0%
Error rate TC_Ad_Add user	2%	0%	-	-	-	1%	-	-

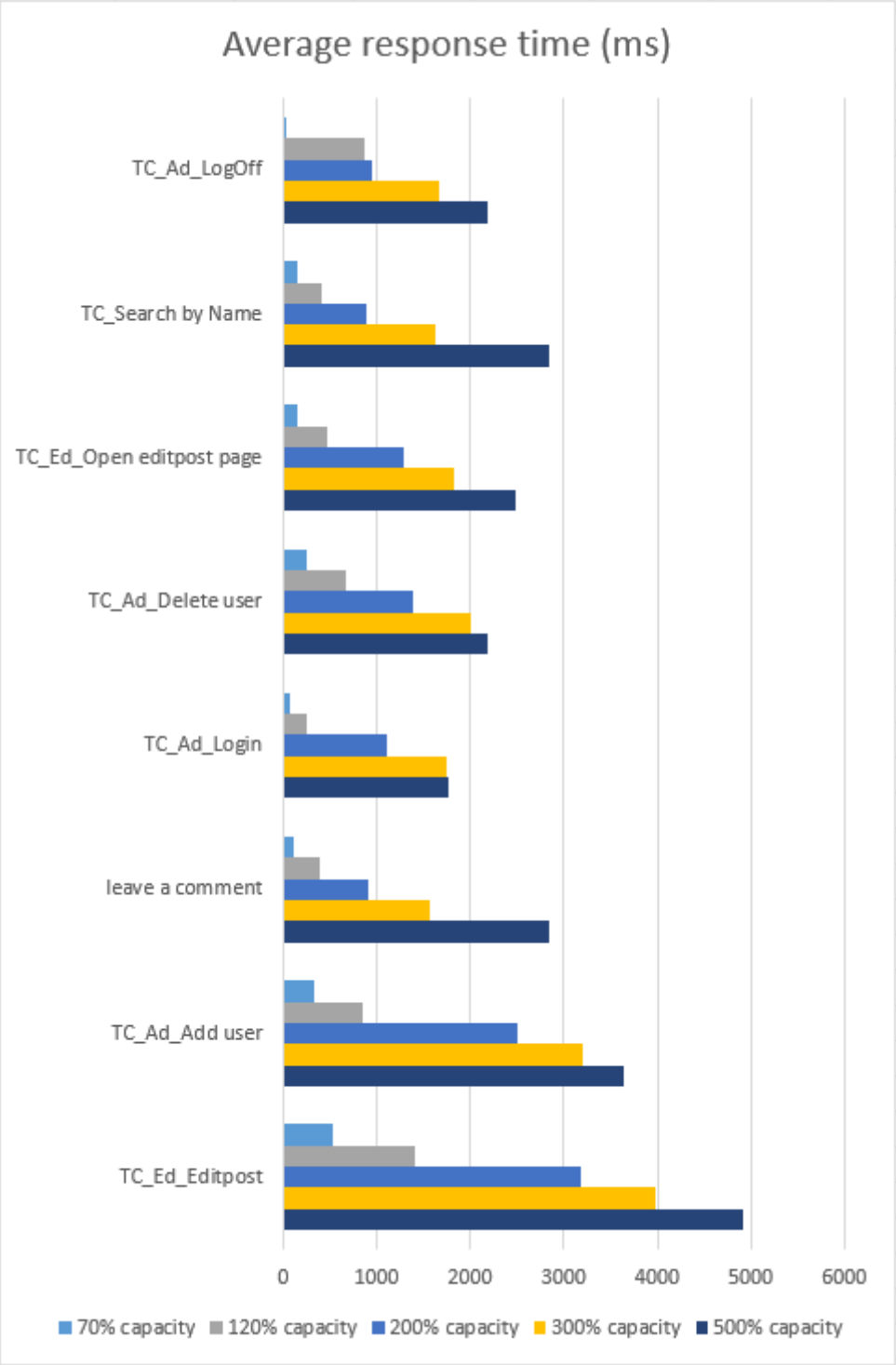
	1x6Gb	DB_1x6Gb	2x6Gb	DB_2x6Gb	3x6Gb	DB_3x6Gb	4x6Gb	DB_4x6Gb	6x6Gb	DB_6x6Gb
Error rate TC_Ed_Editpost	19%	10%	32%	43%	34%	42%	35%	34%	53%	46%
Error rate leave a comment	1%	1%	1%	1%	1%	1%	1%	1%	2%	1%
Error rate Open Random post	-	0%	0%	0%	0%	0%	0%	0%	0%	0%
Error rate TC_Ad_Add user	1%	1%	1%	-	-	0%	1%	0%	-	-

For more details please see "Attachment_1.xlsx".

Results of regular load testing and stress testing:

For clarity, the results of stress testing with a load of 120%, 200%, 300% and 500% system capacity are shown together with the results of regular load testing (70% of the system capacity).

Stress testing was performed for application BlogEngine .NET 3.2 (web) when working with the database as a data source. As a result of stress testing, the system showed a fairly high stability. As the load increased, the average response time was expected to increase, which for the most time-consuming requests could increase ten or more times.



So the average Throughput increased from 30rps (for 70% of the system capacity) to 37-41rps under stress load. At the same time, other key performance Indicators did not change so significantly. An additional confirmation of the stability of the system was the Averge error rate, which under stress load remained in the range of 0.1-0.3%, which is comparable to 0.2% for a system under regular load.

Performance Indicators	70% capacity	120% capacity	200% capacity	300% capacity	500% capacity
Throughput max (rps)	36	58	68	67	87
Throughput avg (rps)	**30**	**38**	**41**	**39**	**37**
CPU usage max (%)	100	100	100	100	100
CPU usage avg (%)	**78**	**90**	**90**	**92**	**92**
Memory usage max (%)	95	89	85	88	88
Memory usage avg (%)	**81**	**86**	**81**	**83**	**83**
IIS stopped during the test	**N**	**N**	**N**	**N**	**N**
An effective duration of test (sec)	**905**	905	**908**	907	**909**
Transactions count	27233	34758	35830	34632	33891
Error count	44	104	114	55	41
Averge error rate (%)	**0.2**	**0.3**	**0.3**	**0.2**	**0.1**

For more details please see "Attachment_2.xlsx".

Volume testing results:

After switching data source from file system to DB volume testing application BlogEngine .NET 3.2 (web) with test data as different numbers of text posts (100, 1000, 2000) and for 1000 posts with a text and attached 1Mb photo showed majority independence key performance indicators on the number of posts and their types existing in the system.

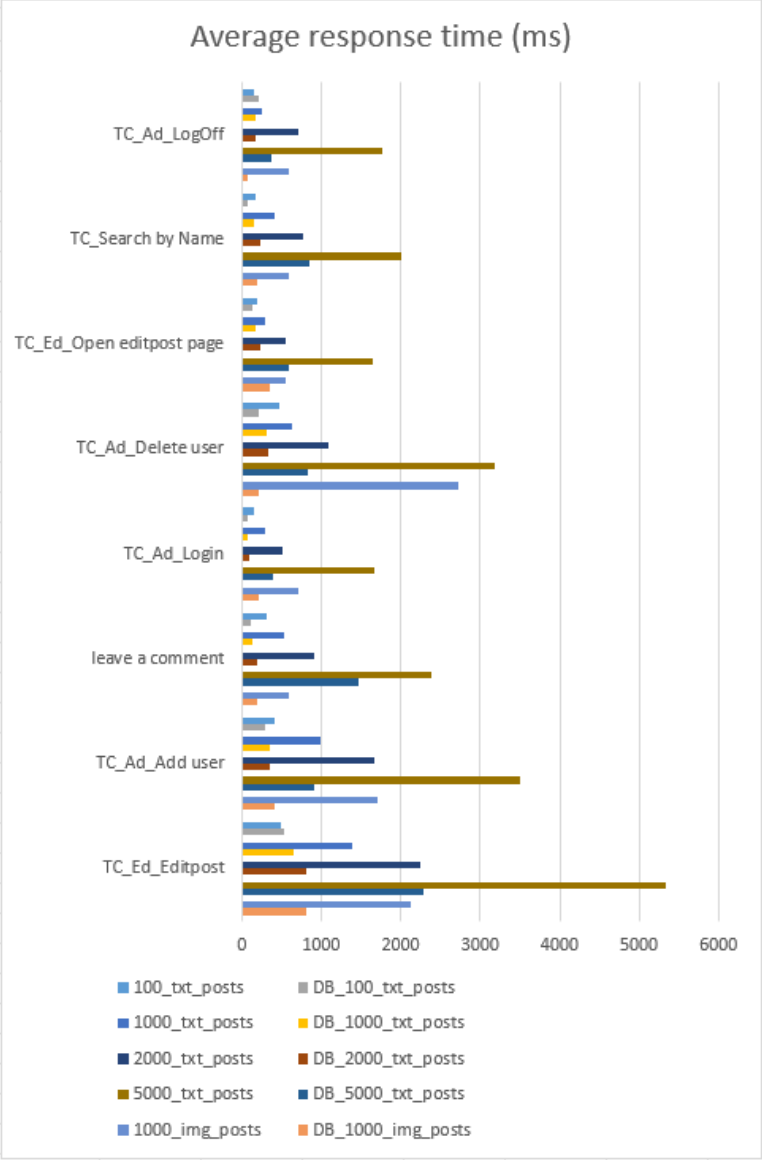
So the average Throughput remained in the range of 29-31 rps and the average CPU usage in the range of 73-82% and the average Memory usage in the range of 82-84%.

Although, as can be seen in the table, when the data source was the file system for the same posts, the range of key performance indicators was much wider. In addition, the average CPU usage was pretty high.

Performance Indicators	100_txt_posts	DB_100_txt_posts	1000_txt_posts	DB_1000_txt_posts	2000_txt_posts	DB_2000_txt_posts	5000_txt_posts	DB_5000_txt_posts	1000_img_posts	DB_1000_img_posts
Throughput max (rps)	46	40	42	37	38	35	25	33	40	37
Throughput avg (rps)	**33**	**31**	**28**	**30**	**22**	**29**	**11**	**20**	**24**	**29**
CPU usage max (%)	100	100	100	100	100	100	100	100	100	100
CPU usage avg (%)	**93**	**73**	**98**	**82**	**97**	**82**	**99**	**97**	**99**	**82**
Memory usage max (%)	90	88	90	88	89	90	85	90	90	89
Memory usage avg (%)	**84**	**82**	**85**	**84**	**84**	**83**	**81**	**83**	**83**	**83**
IIS stopped during the test	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**	**N**
An effective duration of test (sec)	**908**	907	**909**	905	**908**	905	**908**	906	**907**	906
Transactions count	30099	28015	25255	27381	19773	26074	10374	18009	21555	26387
Error count	28	10	86	79	120	133	68	144	69	73
Averge error rate (%)	**0.1**	**0.0**	**0.3**	**0.3**	**0.6**	**0.5**	**0.7**	**0.8**	**0.3**	**0.3**

At the same time, both for the system whose data source is the file system and the database, the indicators for 5000 text posts are significantly different. So the average Throughput for a system whose data source is a file system is reduced by 3 times compared to 100 text posts, but for a system whose data source is a database, the influence is much less, only by one third.

As you can see, the average CPU usage for a system whose data source is a database for 5000 posts rises close to the limit value, namely 97%. Also, for 5000 text posts, a multiple increase in the average response time for the most time-consuming requests and an increase in the average error rate are noticeable.



Error rate: As you can see the Average error rate is less than 1%.

But if you look in more detail in the context of requests, you can see the dependence of the percentage of errors on the number of posts. For a database-sourced system, "Editpost" query errors are 2% for 100 posts, 15% for 1000 posts, 32% for 2000 posts, and 46% for 5000 posts.

Which, as already mentioned in the section Scalability testing results this percentage of errors looks unacceptable in terms of user experience for the "editor" role.

If the decision is made that the data source will be the file system, then attention may also be required by a rather high error rate of the "Add user" request for the admin role, the value of which reached 6% for 2000 and 5000 posts.

For more details please see "Attachment_3.xlsx".

Results of long-time testing:

After switching data source from file system to DB volume testing application BlogEngine .NET 3.2 (web) it was additionally perform long-time testing under low load (~30% of the system capacity) for 8 hours. The main

purpose of this test was to identify possible errors associated with memory leaks and other unexpected behavior associated with the duration and slow growth of data.

Performance Indicators	file_mode	DB_mode
Throughput max (rps)	18	15
Throughput avg (rps)	**12**	**12**
CPU usage max (%)	100	100
CPU usage avg (%)	**86**	**71**
Memory usage max (%)	92	89
Memory usage avg (%)	**81**	**82**
IIS stopped during the test	**N**	**N**
An effective duration of test (sec)	**28807**	28807
Transactions count	350492	330784
Error count	2110	1475
Averge error rate (%)	**0.6**	**0.4**

The application BlogEngine .NET 3.2 (web) database-sourced showed higher mean Throughput stability throughout the test, lower mean CPU usage and mean error rate, and lower deviations in mean Memory usage.

For more details please see "Attachment_4.xlsx".

4 What was tested

Parameters of the system under test are described in paragraph 6.1. Test strategy

5 Load generator system options

Load generator system options are described in paragraph 6.2. Test strategy

6 Test scenarios (scripts)

Detailed description of test scenarios (scripts) are described in paragraph 2.6. Test plan

7 Detailed test results

For more details please see:

"Attachment_1.xlsx"

"Attachment_2.xlsx"

"Attachment_3.xlsx"

"Attachment_4.xlsx"

8 Conclusion

A series of various performance tests have shown that working with DB as a data source, the BlogEngine.NET 3.2 (web) application, supporting the work of a comparable number of concurrent users (see Capacity testing results) as well as a file system as a data source, has significantly higher stability. The confirmation is that IIS (Internet Information Services) doesn't handle with load test with more than one CPU when dealing with the file system as a data source. IIS stops working after some time, 3-5 minutes after the start of the test (see Scalability testing results).

After switching data source from file system to DB volume testing application BlogEngine .NET 3.2 (web) with test data as different numbers of text posts showed majority independence key performance indicators on the number of posts and their types existing in the system. So the average Throughput for a system whose data source is a file system is reduced by 3 times compared 5000 vs 100 text posts, but for a system whose data source is a database, the influence is much less, only by one third (see Volume testing results).

Stress testing was performed for application BlogEngine .NET 3.2 (web) when working with the database as a data source. As a result of stress testing, the system showed a fairly high stability. As the load increased, the average response time was expected to increase, but other key performance Indicators did not change so significantly. An additional confirmation of the stability of the system was the Averge error rate, which under stress load remained in the range of 0.1-0.3%, which is comparable to 0.2% for a system under regular load (see stress testing results).

Error rate: The error rate generally stays under 1%, with the exception of the "Editpost" (TC_Ed_Editpost) request, which is between 10% and 53% of this type of request (see Scalability testing results and Volume testing results).

Considering the totality of the test results, we can conclude that working with DB as a data source for the BlogEngine.NET 3.2 (web) application looks more optimal in terms of performance and stability. At the same time, it makes sense to pay attention to a fairly high percentage of errors on the "Editpost" (TC_Ed_Editpost) request and decide whether this is acceptable for the "editor" role that the customer's staff will encounter.