Ocena efektywność systemów komputerowych –

Porównanie miar szybkości renderowania witryny po stronie serwera i klienta

Autor: Maciej Kraśny 132260

1. Wstęp

Ostatnimi czasy coraz częściej pojawiają się dyskusje na temat renderowania stron internetowych. Programiści biorą udział w wyścigu, w którym głównym celem jest optymalizacja czasu ładowania strony, a co za tym idzie szybkość jej renderowania, ponieważ im szybciej użytkownik będzie mógł zobaczyć oraz wchodzić w interakcję ze stroną, tym lepsze będą jego doświadczenia. Głównymi podjeściami renderowania stron są metody wykonywania tego po stronie serwera lub po stronie klienta.

Renderowanie po stronie serwera to metoda renderowania, w której wszystkie zasoby strony są przechowywane na serwerze. Gdy strona jest żądana, serwer sprawdza zasoby oraz tworzy zawartość pliku HTML. Ten przygotowany kod HTML jest wysyłany do przeglądarki klienta w celu dalszego renderowania i wyświetlania. Następnie przeglądarka pobiera CSS'y, obrazki oraz JavaScript'y , które wykonuje sprawiając, że strona staje się interaktywna.

Renderowanie po stronie klienta to metoda renderowania polegająca na wykonywaniu kodu JavaScript po stronie klienta (przeglądarki) za pośrednictwem JavaScript'u. Klient najpierw zażąda kodu źródłowego, który będzie zawierał niewielką ilość indeksowalnego HTML. Następnie zostanie wysłane drugie żądanie dla plików .js, które służą do zbudowania strony.

Należy sprawdzić, które z tych rozwiązań będzie zapewniać szybsze renderowanie strony w zależności od obciążenia serwera przez wielu równoległych użytkowników.

2. Eksperyment

2.1 Opis

Do przeprowadzenia eksperymentu zostały napisane dwie aplikacje internetowe przy użyciu biblioteki JavaScript'owej React, które dostarczają użytkownikowi ten sam kontent. Do napisania pierwszej został użyty React'owy framework NextJS, który pozwala na stworzenie aplikacji React'owej z renderowaniem po stronie serwera. Natomiast do napisanie drugiej zostało wykorzystane narzędzie create-react-app, które pozwala na tworzenie projektu opartego na React. Tworzy strukturę katalogów i plików oraz zawiera wszystkie potrzebne na początek narzędzia. Reprezentuje ona sytuacje renderowania aplikacji po stronie klienta.

2.2 Narzędzia

Do stworzenia środowiska, w którym serwer jest obciążony przez wielu równoległych użytkowników została wykorzystana biblioteka Autocannon napisana w NodeJS.
Do wykonania pomiarów zostało użyte narzędzie Lighthouse, który umożliwia podgląd wydajności, jakości oraz poprawności aplikacji internetowych.

2.3 Miary

First contentful paint, to czas, w którym przeglądarka renderuje pierwszą część treści z DOM, po przejściu użytkownika na stronę.

Large contentful paint, to czas renderowania największego obrazu lub bloku tekstu widocznego w aktualnym obszarze strony.

Time to interactive, to miara, która mówi, ile czasu minie, aby strona stała się w pełni interaktywna.

2.4 Przebieg eksperymentu

Dla liczby użytkowników 1, 10, 100, 200, 300, 400, 500, 650, 800 zostało uruchomione narzędzie autocannon komendą autocannon -c x -d 60 localhost:3000, gdzie x jest liczbą równoległych użytkowników. Generowało to przez 60 sekund obciążenie na serwerze dla zadanej liczby użytkowników. W tym samym czasie przy użyciu programu lighthouse generowany był raport. Dla każdej instancji pomiar był wykonywany 5-krotnie. Serwer został uruchomiony na lokalnej maszynie.

Parametry maszyny lokalnej:

Dodatkowo został przeprowadzony test dla 100 równoległych użytkowników przy obciążeniu trwającym 60s, a aplikacja została wdrożona na maszynie wirtualnej:

Parametry maszyny wirtualnej:

CPU: VCPU Latest AMD
RAM: 1GB
SWAP: 2GB
Pojemność dysku: 25 GB
System operacyjny: CentOS 8

Parametry maszyny lokalnej:

CPU: Intel i7-8850H
RAM: 16GB
SWAP: 0
Pojemność dysku: 256 GB
System operacyjny: Windows 10 Pro

3. Wyniki

Ze względu na to, że wyniki pomiarów dla tej samej liczby użytkowników są do siebie zbliżone oraz nie ma krytycznych różnic jako wartość centralna została zastosowana średnia arytmetyczna. Wszystkie wyniki pomiarów wyrażone są w sekundach.

3.1 First contentful paint

Tabela 1 Wyniki eksperymentu dla miary FCP przy renderowaniu po stronie klienta

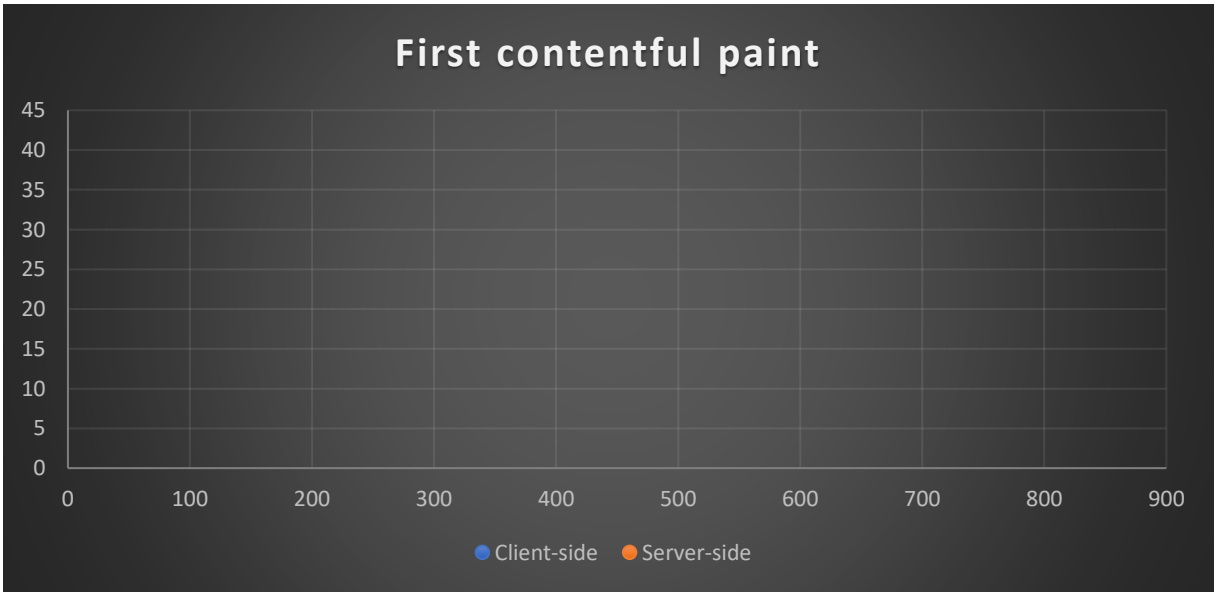
Nr\L. użytkowników	**1**	**10**	**50**	**100**	**200**	**300**	**400**	**500**	**650**	**800**
1	1,4	1,5	1,6	1,6	1,6	2	2,7	2,8	2,8	3,1
2	1,3	1,5	1,6	1,7	2,1	2,2	3	2,8	2,6	2,9
3	1,4	1,5	1,6	1,9	2,1	2,3	2,6	3	2,8	3
4	1,3	1,6	1,6	1,6	2	2,1	3	2,9	2,9	2,9
5	1,6	1,5	1,7	1,7	1,9	2,3	2,7	3	3	3,1
Odch. stand.	0,12	0,04	0,04	0,12	0,21	0,13	0,19	0,1	0,15	0,1
Śr. arytm.	1,4	1,52	1,62	1,7	1,94	2,18	2,8	2,9	2,82	3

Średnia arytmetyczna dla FCP przy 100 równoległych użytkownikach dla strony wdrożonej na maszynie wirtualnej i renderowaniu po stronie klienta: 2,8

Tabela 2 Wyniki eksperymentu dla miary FCP przy renderowaniu po stronie serwera

Nr\L. użytkowników	1	10	50	100	200	300	400	500	650	800
1	0,8	0,9	2,5	6,1	10,9	14,1	20,1	24,5	33,4	41,5
2	0,7	1	3	6,1	11,1	13,8	20,4	24,5	33	41,6
3	0,8	1,1	2,9	6	11,4	14,1	19,9	26	33,5	41,7
4	0,6	0.9	3	6	11,3	13,9	20,3	25	33,1	41,7
5	0,7	1,1	2,8	5,9	11,1	13,8	20,3	24,9	33,5	42
Odch. stand.	0,08	0,47	0,207	0,083	0,194	0,151	0,2	0,614	0,235	0,187
Śr. arytm.	0,72	0,82	2,84	6,02	11,16	13,94	20,2	24,98	33,3	41,7

Średnia arytmetyczna dla FCP przy 100 równoległych użytkownikach dla strony wdrożonej na maszynie wirtualnej i renderowaniu po stronie serwera: 1,9



Rysunek 1 Wyniki eksperymentu dla miary FCP

3.2 Large contentful paint

Tabela 3 Wyniki eksperymentu dla miary LCP przy renderowaniu po stronie klienta

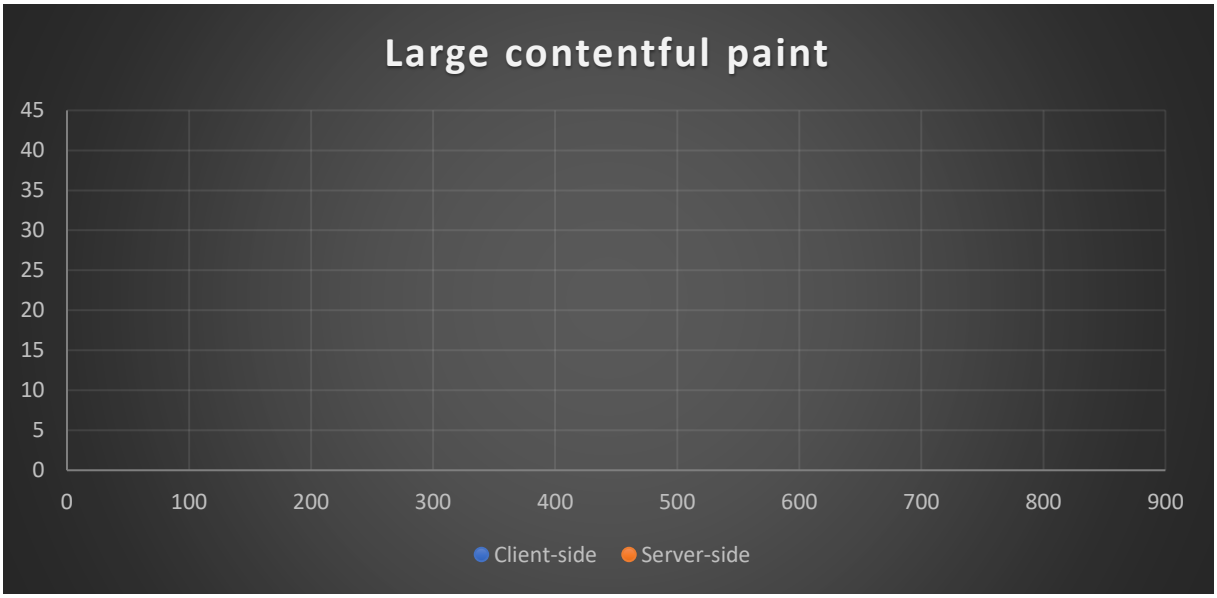
Nr\L. użytkowników	1	10	50	100	200	300	400	500	650	800
1	1,8	1,9	2,1	2,1	2,1	2,3	3	3	3	3,3
2	1,7	1,9	2	2	2,2	2,2	3,4	2,9	3,1	2,9
3	1,8	1,9	2,2	2,2	2,1	2,4	3	3,1	3,2	3,3
4	1,8	2	2,2	2,1	2,1	2,2	3,2	3,1	3,1	3,3
5	1,7	1,9	2	2	2,2	2,4	3,2	3,2	3,2	3,3
Odch. stand.	0,054	0,04	0,1	0,084	0,055	0,1	0,167	0,114	0,084	0,179
Śr. arytm.	1,76	1,92	2,1	2,08	2,14	2,3	3,16	3,06	3,12	3,22

Średnia arytmetyczna dla FCP przy 100 równoległych użytkownikach dla strony wdrożonej na maszynie wirtualnej i renderowaniu po stronie klienta: 3,0

Tabela 4 Wyniki eksperymentu dla miary LCP przy renderowaniu po stronie serwera

Nr\L. użytkowników	1	10	50	100	200	300	400	500	650	800
1	0,8	0,9	2,5	6,1	10,9	14,1	20,1	24,5	33,4	41,5
2	0,7	1	3	6,1	11,1	13,8	20,4	24,5	33	41,6
3	0,8	1,1	2,9	6	11,4	14,1	19,9	26	33,5	41,7
4	0,6	0.9	3	6	11,3	13,9	20,3	25	33,1	41,7
5	0,7	1,1	2,8	5,9	11,1	13,8	20,3	24,9	33,5	42
Odch. stand.	0,08	0,46	0,207	0,084	0,195	0,152	0,2	0,614	0,235	0,187
Śr. arytm.	0,72	0,82	2,84	6,02	11,16	13,94	20,2	24,98	33,3	41,7

Średnia arytmetyczna dla FCP przy 100 równoległych użytkownikach dla strony wdrożonej na maszynie wirtualnej i renderowaniu po stronie klienta: 2,0



Rysunek 2 Wyniki eksperymentu dla miary LCP

3.3 Time to interactive

Tabela 5 Wyniki eksperymentu dla miary TTI przy renderowaniu po stronie klienta

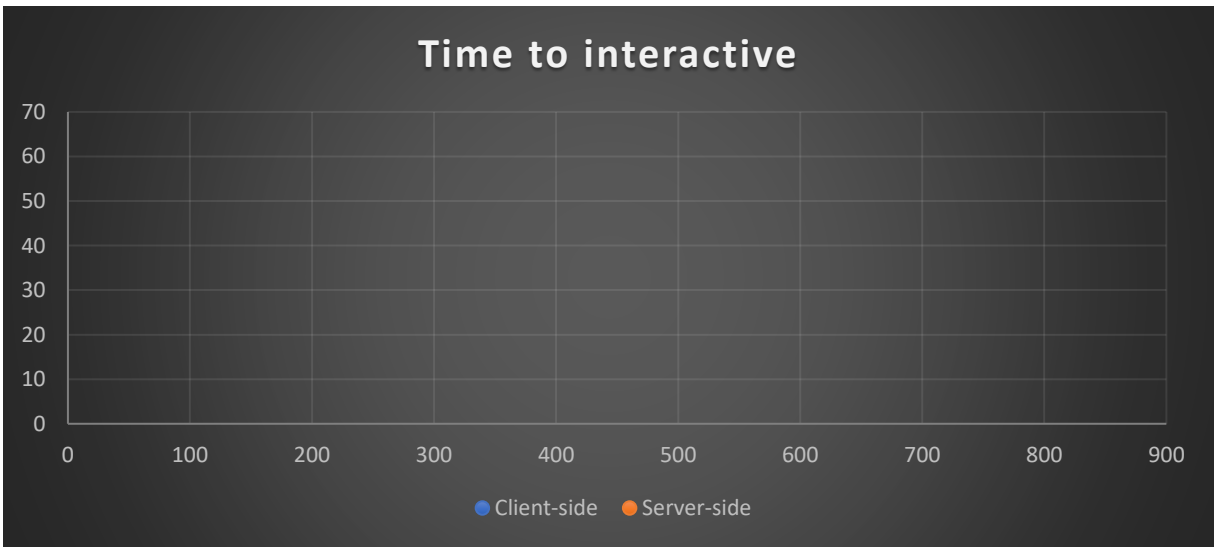
Nr\L. użytkowników	1	10	50	100	200	300	400	500	650	800
1	3,1	3,1	3,2	3,2	3,2	3,2	3,4	3,4	3,5	3,8
2	3	3,1	3,1	3	3,3	3,5	3,2	3,4	3,3	3,7
3	3,1	3,3	3,2	2,9	3,4	3,6	3,4	3,8	3,5	3,8
4	3,2	3,1	3,3	3	3,5	3,3	3,5	3,7	3,5	3,7
5	3,1	3,3	3,2	3,3	3,4	3,6	3,4	3,7	3,6	3,9
Odch. stand.	0,07	0,11	0,071	0,164	0,114	0,182	0,11	0,187	0,11	0,084
Śr. arytm.	3,1	3,18	3,2	3,08	3,36	3,44	3,38	3,6	3,48	3,78

Średnia arytmetyczna dla FCP przy 100 równoległych użytkownikach dla strony wdrożonej na maszynie wirtualnej i renderowaniu po stronie klienta: 3,3

Tabela 6 Wyniki eksperymentu dla miary TTI przy renderowaniu po stronie serwera

Nr\L. użytkowników	1	10	50	100	200	300	400	500	650	800
1	1,6	1,7	4	8,7	15,3	23,6	29,9	34,8	46,1	57
2	1,6	1,8	4,3	8,8	15,8	23,4	28,7	35	46,9	57,1
3	1,6	2	4,5	8,5	15,9	23,6	29	35,6	46,6	57,4
4	1,5	1,9	4,2	8,7	15,4	23,6	29,5	34,8	46,15	57,1
5	1,6	1,8	4,3	8,6	15,8	23,5	29,1	35,1	46,7	57,2
Odch. stand.	0,04	0,11	0,182	0,114	0,27	0,089	0,467	0,329	0,351	0,152
Śr. arytm.	1,58	1,84	4,26	8,66	15,64	23,54	29,24	35,06	46,49	57,16

Średnia arytmetyczna dla FCP przy 100 równoległych użytkownikach dla strony wdrożonej na maszynie wirtualnej i renderowaniu po stronie klienta: 3,0



Rysunek 3 Wyniki eksperymentu dla miary TTI

3.4 Obciążenie

Testy obciążeniowe na maszynie wirtualnej wykazały, że renderowanie po stronie klienta może obsłużyć max 180 zapytań, a po stronie serwera 100 bez błędów. Powyżej tych wartości pojawiały się zapytania niepoprawnie obsłużone.

3. Wnioski

Badanie miar szybkości renderowania witryny po stronie serwera i klienta wykazało, że są one zależne od liczby użytkowników, jednak w różnych stopniach, ponieważ większe obciążenie serwera dużo bardziej wpływa na renderowanie po stronie serwera niż klienta. Jest to spowodowane tym, że w pierwszym rozwiązaniu serwer musi wykonać znacznie więcej operacji, gdyż serwer musi przygotować gotowego HTML'a i przesłać go klientowi. W drugiej metodzie tą operację wykonuje przeglądarka użytkownika, czym odciąża serwer. Pomiary wykonane na serwerze uruchomionym lokalnie wykazały, że tylko w przypadku 1 i 10 równoległych użytkowników metoda renderowania po stronie serwera ma mniejsze wartości dla wszystkich miar niż metoda renderowania po stronie klienta, jednak każdy kolejny pomiar wykazał, że pierwsze rozwiązanie nie radzi sobie z większą ilością równoległych użytkowników, a przy 850 należy czekać prawie minutę, aby można było korzystać ze strony internetowej. W tym samym przypadku renderowanie po stronie klienta nadal działa w sposób akceptowalny dla użytkowników, gdyż strona nadal ładuje się poniżej 4s, więc użytkownik prawdopodobnie pozostałby na stronie. Na takie tragiczne wyniki dla renderowania po stronie serwera wpłynęło to, że serwer został uruchomiony na lokalnej maszynie z wymagającym systemem operacyjnym Windows. Serwer musiał również współdzielić zasoby z innymi programami jak na przykład antywirus, a do przeprowadzenia testu musiała być uruchomiona przeglądarka z programem do badania efektywności strony Lighthouse. Te wszystkie programy spowodowały znaczne obniżenie wydajności serwera i dlatego wartości miar były bardzo duże, przy niewielkich różnicach w rednerowaniu po stronie klietna.

Test przy 100 równoległych użytkownikach został również przeprowadzony na serwerze postawionym na dość taniej (5$ miesięcznie) maszynie wirtualnej, a wyniki pokazały, że strona redneruje się szybciej, gdy ta operacja jest wykonywana po stronie serwera niż po stronie klienta.

Testy obciążeniowe wykazały, że serwer może obsłużyć prawie 2 razy mniej zapytań, gdy strona jest renderowana po stronie serwera niż klienta. Jest to zgodne z intuicją, gdyż w przypadku pierwszej metody operacje na serwerze są bardziej złożone i bardziej obciążają serwer powodując to, że mniej zapytań mogą obsłużyć.

Podsumowując, na szybkość rednerowania witryny po stronie klienta ilość użytkowników bardziej wpływa niż w przypadku renderowania po stronie klienta, jednak zastosowanie podstawowej maszyny wirtualnej pokazuję, że szybkość rednerowania strony jest większa, a co za tym idzie czas wyświetlenia kontentu jest mniejszy.