Test numer 1:
Aplikacja: app.py

Testowany endpoint: / oraz /hello

Liczba użytkowników: 1

Tempo generowania użytkowników: 5 na sekundę

Czas trwania testu: 3 minuty

Wyniki:

Liczba żądań: 101

Liczba błędów: 0

Czas odpowiedzi (średni):

/:

Mediana: 8 ms

Czas odpowiedzi dla 90% próbek: 10 ms

Czas odpowiedzi dla 99% próbek: 12 ms

/hello:

Mediana: 8 ms

Czas odpowiedzi dla 90% próbek: 11 ms

Czas odpowiedzi dla 99% próbek: 20 ms

Rozkład czasu odpowiedzi:

/:

Minimum: 7 ms

Maksimum: 12 ms

/hello:

Minimum: 8 ms

Maksimum: 20 ms

Aktualne tempo żądań na sekundę (RPS): 0.5

Aktualna liczba błędów na sekundę: 0

Podsumowanie:

Na podstawie przeprowadzonych testów z użyciem Locust na aplikacji testowej z jednym użytkownikiem i generowaniu 5 użytkowników na sekundę przez okres 3 minut, można stwierdzić, że:

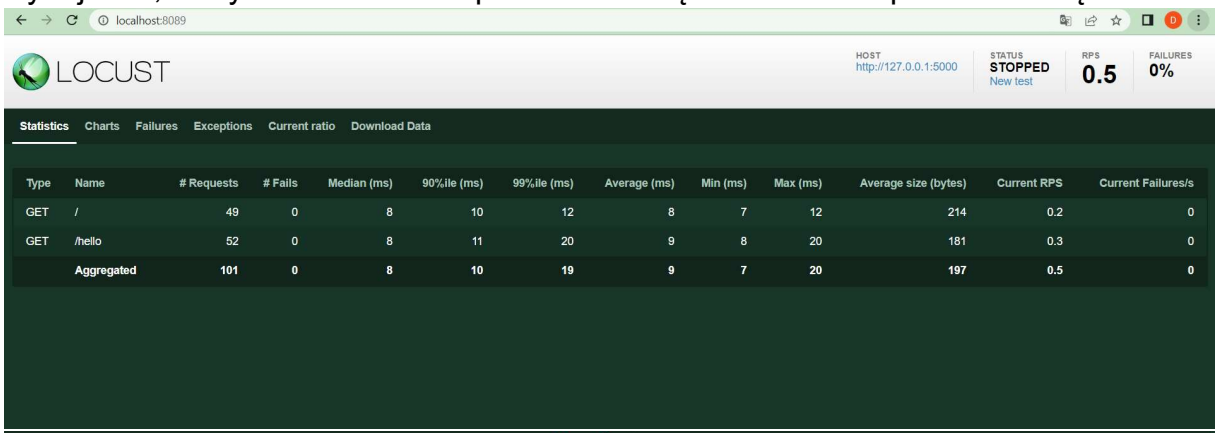
Czas odpowiedzi endpointu / był zazwyczaj poniżej 10 ms, z mediana wynoszącą 8 ms.

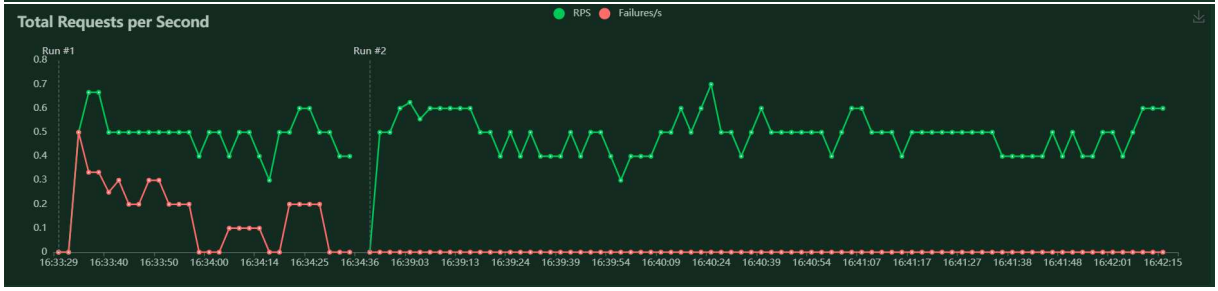
Czas odpowiedzi endpointu /hello był zazwyczaj poniżej 11 ms, z mediana wynoszącą 8 ms.

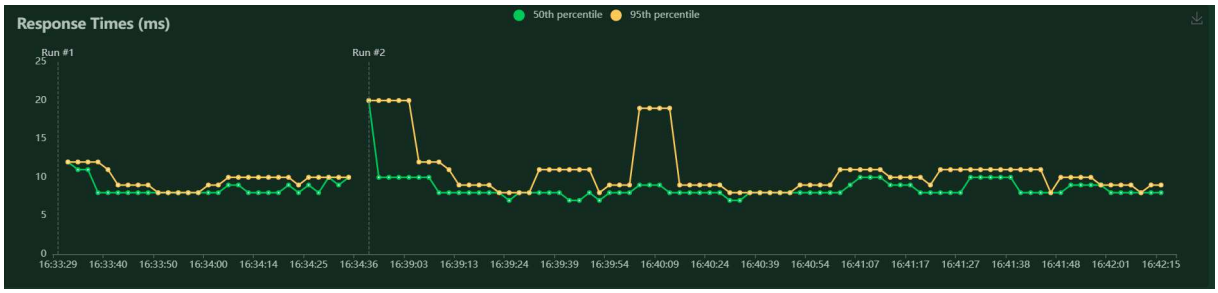
Wystąpiły bardzo niewielkie opóźnienia w czasie odpowiedzi, z czasem odpowiedzi dla 90% próbek wynoszącym maksymalnie 12 ms dla / i 20 ms dla /hello.

Nie wystąpiły żadne błędy podczas testów.

W oparciu o te wyniki, można stwierdzić, że aplikacja testowa wykazuje dobrą wydajność, z szybkim czasem odpowiedzi na żądania i niskim poziomem błędów.







Test numer 2
Aplikacja: app.py

Testowany endpoint: / oraz /hello

Liczba użytkowników: 50

Tempo generowania użytkowników: 1 na sekundę

Czas trwania testu: 1 minuta

Wyniki:

Liczba żądań: 949

Liczba błędów: 0

Czas odpowiedzi (średni):

/:

Mediana: 8 ms

Czas odpowiedzi dla 90% próbek: 13 ms

Czas odpowiedzi dla 99% próbek: 19 ms

/hello:

Mediana: 9 ms

Czas odpowiedzi dla 90% próbek: 13 ms

Czas odpowiedzi dla 99% próbek: 20 ms

Aktualne tempo żądań na sekundę (RPS): 24.7

Aktualna liczba błędów na sekundę: 0

Podsumowanie:

Na podstawie przeprowadzonych testów z użyciem Locust na aplikacji testowej z 50 użytkownikami o generowaniu 1 użytkownika na sekundę przez okres 1 minuty, można stwierdzić, że:

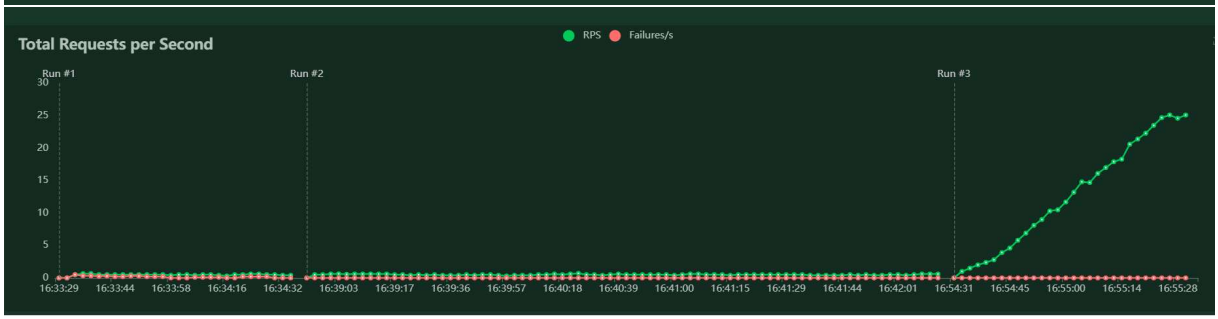
Czas odpowiedzi endpointu / był zazwyczaj poniżej 13 ms, z mediana wynoszącą 8 ms.

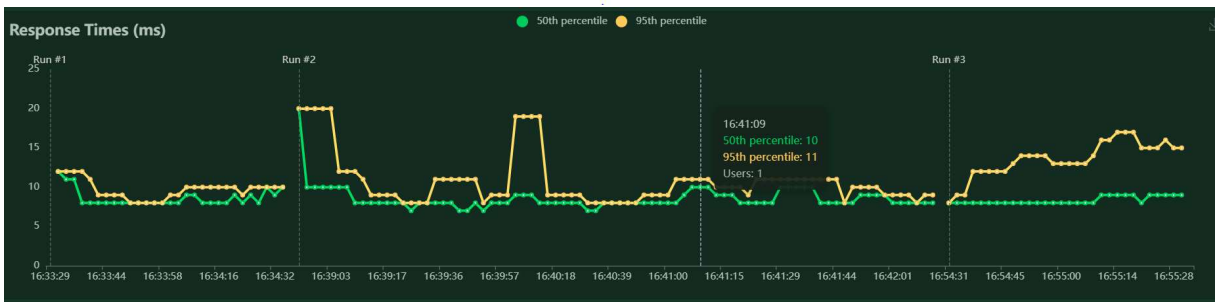
Czas odpowiedzi endpointu /hello był zazwyczaj poniżej 13 ms, z mediana wynoszącą 9 ms.

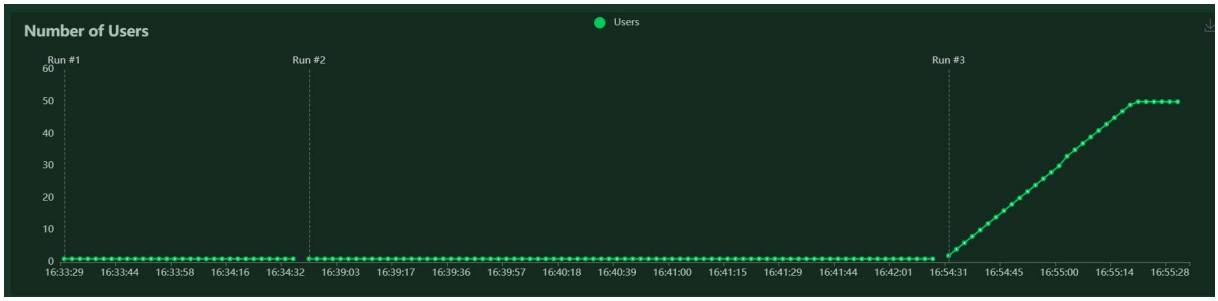
Wystąpiły niewielkie opóźnienia w czasie odpowiedzi, z czasem odpowiedzi dla 90% próbek wynoszącym maksymalnie 19 ms dla / i 20 ms dla /hello.

Nie wystąpiły żadne błędy podczas testów.W oparciu o te wyniki, można stwierdzić, że aplikacja testowa wykazuje dobrą wydajność, z szybkim czasem odpowiedzi na żądania i niskim poziomem błędów, nawet przy większym obciążeniu serwera generowanym przez 50 użytkowników.

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	/	468	0	8	13	19	9	7	23	214	13.4	0
GET	/hello	481	0	9	13	20	10	8	70	181	11.3	0
	Aggregated	949	0	8	13	19	10	7	70	197	24.7	0







Test numer 3:

Aplikacja testowa: App.py

Testowany endpoint: / oraz /hello

Liczba użytkowników: 500

Tempo generowania użytkowników: 1 na sekundę

Czas trwania testu: 1 minuta

Wyniki:

Liczba żądań: 1750

Liczba błędów: 0

Czas odpowiedzi (średni):

/:

Mediana: 9 ms

Czas odpowiedzi dla 90% próbek: 15 ms

Czas odpowiedzi dla 99% próbek: 23 ms

/hello:

Mediana: 10 ms

Czas odpowiedzi dla 90% próbek: 16 ms

Czas odpowiedzi dla 99% próbek: 24 ms

Aktualne tempo żądań na sekundę (RPS): 37.4

Aktualna liczba błędów na sekundę: 0

Podsumowanie:

Na podstawie przeprowadzonych testów z użyciem Locust na aplikacji testowej z 500 użytkownikami o generowaniu 1 użytkownika na sekundę przez okres 1 minuty, można stwierdzić, że:

Czas odpowiedzi endpointu / był zazwyczaj poniżej 15 ms, z mediana wynoszącą 9 ms.

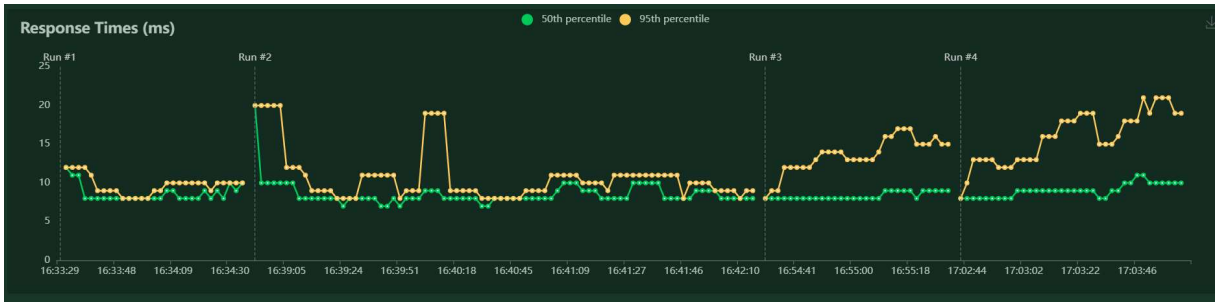
Czas odpowiedzi endpointu /hello był zazwyczaj poniżej 16 ms, z mediana wynoszącą 10 ms.

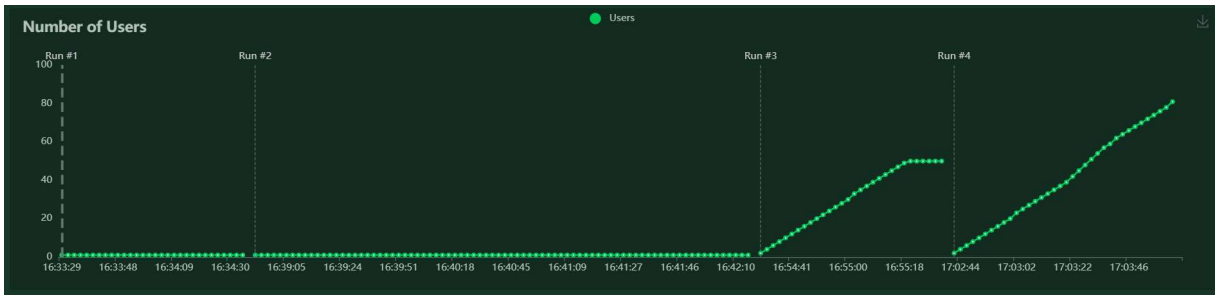
Wystąpiły pewne opóźnienia w czasie odpowiedzi, z czasem odpowiedzi dla 90% próbek wynoszącym maksymalnie 23 ms dla / i 24 ms dla /hello.

Nie wystąpiły żadne błędy podczas testów.

W oparciu o te wyniki, można stwierdzić, że aplikacja testowa nadal wykazuje dobrą wydajność, chociaż czas odpowiedzi nieco wzrósł w porównaniu do testu z mniejszą liczbą użytkowników (50). Obciążenie serwera generowane przez 500 użytkowników nie spowodowało wystąpienia błędów, a aplikacja była w stanie obsłużyć takie obciążenie z zadowalającym czasem odpowiedzi.

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	/	902	0	9	15	23	10	6	76	214	19.1	0
GET	/hello	848	0	10	16	24	11	7	77	181	18.3	0
	Aggregated	**1750**	**0**	**9**	**16**	**23**	**11**	**6**	**77**	**198**	**37.4**	**0**





Test numer 4:

Aplikacja testowa: app.py

Testowany endpoint: / oraz /hello

Liczba użytkowników: 1000

Tempo generowania użytkowników: 1 na sekundę

Czas trwania testu: 2 minuty

Wyniki:

Liczba żądań: 3674

Liczba błędów: 0

Czas odpowiedzi (średni):

/:

Mediana: 10 ms

Czas odpowiedzi dla 90% próbek: 18 ms

Czas odpowiedzi dla 99% próbek: 29 ms

/hello:

Mediana: 11 ms

Czas odpowiedzi dla 90% próbek: 19 ms

Czas odpowiedzi dla 99% próbek: 32 ms

Aktualne tempo żądań na sekundę (RPS): 56.4

Aktualna liczba błędów na sekundę: 0

Podsumowanie:

Na podstawie przeprowadzonych testów z użyciem Locust na aplikacji testowej z 1000 użytkownikami o generowaniu 1 użytkownika na sekundę przez okres 2 minut, można stwierdzić, że:

Czas odpowiedzi endpointu / był zazwyczaj poniżej 18 ms, z medianą wynoszącą 10 ms.

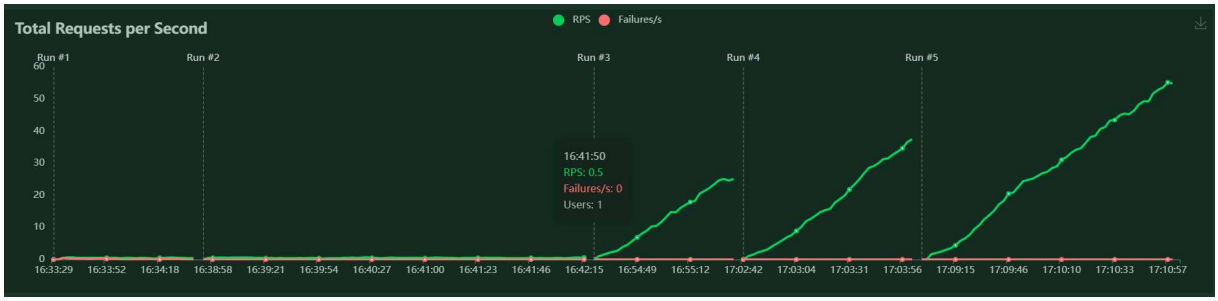
Czas odpowiedzi endpointu /hello był zazwyczaj poniżej 19 ms, z medianą wynoszącą 11 ms.

Wystąpiły pewne opóźnienia w czasie odpowiedzi, z czasem odpowiedzi dla 90% próbek wynoszącym maksymalnie 29 ms dla / i 32 ms dla /hello.

Nie wystąpiły żadne błędy podczas testów.

W oparciu o te wyniki, można stwierdzić, że aplikacja testowa nadal wykazuje zadowalającą wydajność, chociaż czas odpowiedzi nieco wzrósł w porównaniu do testów z mniejszą liczbą użytkowników. Obciążenie serwera generowane przez 1000 użytkowników nie spowodowało wystąpienia błędów, a aplikacja była w stanie obsłużyć takie obciążenie z akceptowalnym czasem odpowiedzi.

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	/	1829	0	10	18	29	11	6	44	214	27.3	0
GET	/hello	1845	0	11	19	32	12	7	90	181	29.1	0
	Aggregated	**3674**	**0**	**10**	**18**	**31**	**12**	**6**	**90**	**197**	**56.4**	**0**



Test numer 5:

Aplikacja testowa: app.py

Testowany endpoint: / oraz /hello

Liczba użytkowników: 5000

Tempo generowania użytkowników: 1 na sekundę

Czas trwania testu: 3 minuty

Wyniki:

Liczba żądań: 10065

Liczba błędów: 0

Czas odpowiedzi (średni):

/:

Mediana: 6 ms

Czas odpowiedzi dla 90% próbek: 12 ms

Czas odpowiedzi dla 99% próbek: 19 ms

/hello:

Mediana: 7 ms

Czas odpowiedzi dla 90% próbek: 12 ms

Czas odpowiedzi dla 99% próbek: 19 ms

Aktualne tempo żądań na sekundę (RPS): 96.7

Aktualna liczba błędów na sekundę: 0

Podsumowanie:

Na podstawie przeprowadzonych testów z użyciem Locust na aplikacji testowej z 5000 użytkownikami o generowaniu 1 użytkownika na sekundę przez okres 3 minut, można stwierdzić, że:

Czas odpowiedzi endpointu / był zazwyczaj poniżej 12 ms, z medianą wynoszącą 6 ms.

Czas odpowiedzi endpointu /hello był zazwyczaj poniżej 12 ms, z medianą wynoszącą 7 ms.

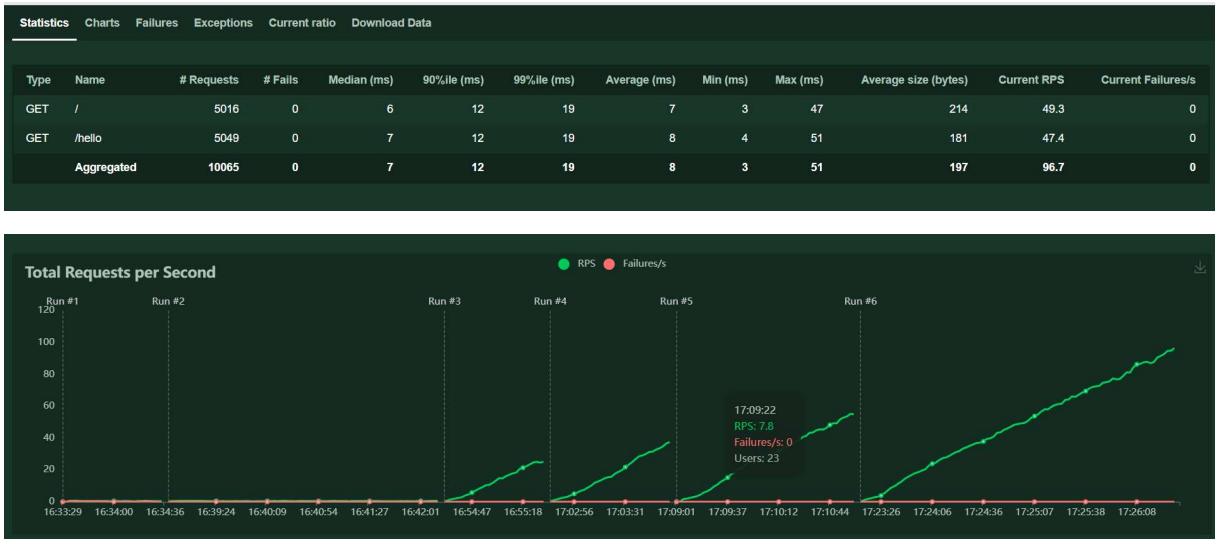
Wystąpiły pewne opóźnienia w czasie odpowiedzi, z czasem odpowiedzi dla 90% próbek wynoszącym maksymalnie 19 ms dla / i /hello.

Nie wystąpiły żadne błędy podczas testów.

W oparciu o te wyniki, można stwierdzić, że aplikacja testowa wykazuje zadowalającą wydajność nawet przy dużym obciążeniu generowanym przez 5000 użytkowników. Czasy odpowiedzi były akceptowalne, a liczba błędów była równa 0, co wskazuje na stabilną i wydajną pracę aplikacji w warunkach takiego obciążenia.

Statistics | Charts | Failures | Exceptions | Current ratio | Download Data

Type	Name	# Requests	# Fails	Median (ms)	90%ile (ms)	99%ile (ms)	Average (ms)	Min (ms)	Max (ms)	Average size (bytes)	Current RPS	Current Failures/s
GET	/	5016	0	6	12	19	7	3	47	214	49.3	0
GET	/hello	5049	0	7	12	19	8	4	51	181	47.4	0
	Aggregated	**10065**	**0**	**7**	**12**	**19**	**8**	**3**	**51**	**197**	**96.7**	**0**



Podsumowując wyniki pięciu raportów z testów Locust na aplikacji app.py, można stwierdzić, że aplikacja wykazuje stabilną i wydajną pracę przy zwiększającym się obciążeniu. Czasy odpowiedzi są akceptowalne, a liczba błędów wynosi 0. Aktualne tempo żądań na sekundę (RPS) jest na akceptowalnym poziomie, co wskazuje na zdolność aplikacji do obsługi większej liczby użytkowników jednocześnie.