Filip Kaczmarek 139949
Andrzej Kapczyński 145358
Patryk Jędrzejewski 145221

Gymshare

Opis aplikacji

Gymshare to aplikacja, która ułatwia zarządzanie treningiem oraz umożliwia udostępnienie go innym. Głównym założeniem systemu jest tworzenie własnego planu treningowego na podstawie bazy dostępnych ćwiczeń, którą zarządza administrator. Utworzony plan może być prywatny lub publicznie dostępny dla wszystkich, w zależności od decyzji autora. Użytkownik ma mieć możliwość odtworzenia treningu, dzięki interfejsowi symulacji, który przeprowadzi użytkownika w odpowiedniej kolejności przez każde ćwiczenie oraz zapisze postępy w sekcji statystyk. Program ma służyć użytkownikom lokalnej siłowni znajdującej się przy ul. Wioślarskiej w Poznaniu.

Aplikacja została zaprojektowana jako aplikacja mobilna, lecz na potrzeby przedmiotu została przebudowana do aplikacji internetowej. Stos technologiczny: Python, Django, Dart, Flutter, PostgreSQL

Działanie aplikacji

Wymagania funkcjonalne

Niezalogowani oraz nowi użytkownicy aplikacji mają możliwość:

- U1. Stworzenia nowego konta.
- U2. Zresetowania hasła do istniejącego już konta.
- U3. Zalogowania się do aplikacji, podając adres e-mail oraz hasło.

Zalogowany użytkownik ma możliwość:

- Z1. Przeglądania planów treningowych.
- Z2. Tworzenia planów treningowych.
- Z3. Przeglądania listy ćwiczeń.
- Z4. Przeglądania prywatnych statystyk.
- Z5. Edycji danych w profilu.
- Z6. Zmiany hasła do konta.
- Z7. Edycji swoich planów treningowych.
- Z8. Usuwania swoich planów treningowych.
- Z9. Dodawania planów treningowych do listy ulubionych.
- Z10. Oceniania planów treningowych.
- Z11. Uruchamiania planów treningowych..
- Z12. Wylogowania z konta.

Wymagania pozafunkcjonalne

- Aplikacja powinna uruchamiać się na przeglądarkach internetowych Chrome, Opera, Safari, Firefox, Edge.
- Aplikacja powinna mieć dostęp do Internetu.
- Hasła użytkowników w bazie danych powinny być szyfrowane.
- Przewidywana maksymalna liczba zalogowanych użytkowników na raz: 50.
- Maksymalna ilość zarejestrowanych użytkowników: 500.

Testy

Testy zostały podzielone na dwie części. Aplikacja serwerowa zostanie przetestowana narzędziem, za pomocą którego dokonamy analizy wydajnościowej aplikacji przy różnym obciążeniu spowodowanym coraz większą liczbą aktywnych użytkowników. W przypadku aplikacji internetowej zostaną wykonane testy integracyjne co pozwoli sprawdzić jak aplikacja zachowuje się od strony użytkownika w przypadku różnego jej obciążenia. Testy zostaną wykonane przed i po optymalizacji aby porównać wyniki w celu weryfikacji jej jakości.

Funkcje do testowania i oczekiwane czasy odpowiedzi aplikacji serwerowej.

Nazwa funkcji	**Czas odpowiedzi [s]**
Tworzenie konta	2
Logowanie na konto użytkownika	2
Utworzenie treningu	4
Wyświetlenie listy treningów	2
Przetwarzanie zakończenia treningu	1
Wystawienie oceny treningu	1
Czas generowania statystyk spalonych kalorii	3
Czas generowania statystyk odbytych ćwiczeń	3

Scenariusze testowania aplikacji internetowej

1. Tworzenie konta
    a. Użytkownik przechodzi do strony rejestracji.
    b. Użytkownik wypełnia formularz a następnie go zatwierdza.
    c. Użytkownik loguje się do aplikacji po uzyskaniu informacji o założeniu konta.

2. Wyświetlenie treningów
    a. Użytkownik loguje się do aplikacji.
    b. Użytkownik przechodzi do zakładki “Home”.
    c. Użytkownik wyświetla 30 treningów.

3. Wyświetlenie statystyk spalonych kalorii
    a. Użytkownik loguje się do aplikacji.
    b. Użytkownik przechodzi do zakładki “Statistics”.
    c. Użytkownik wybiera datę.
    d. Użytkownik wyświetla wygenerowany wykres.

4. Wyświetlenie statystyk wykonanych ćwiczeń
    a. Użytkownik loguje się do aplikacji.
    b. Użytkownik przechodzi do zakładki “Statistics”.
    c. Użytkownik zmienia górną zakładkę na “Exercises”.
    d. Użytkownik wybiera datę.
    e. Użytkownik wyświetla historię ćwiczeń.

Populowanie bazy danych

Baza danych została wypełniona testowymi danymi, tak aby spełniać przybliżone warunki przy 500 zarejestrowanych użytkownikach. Dodano:

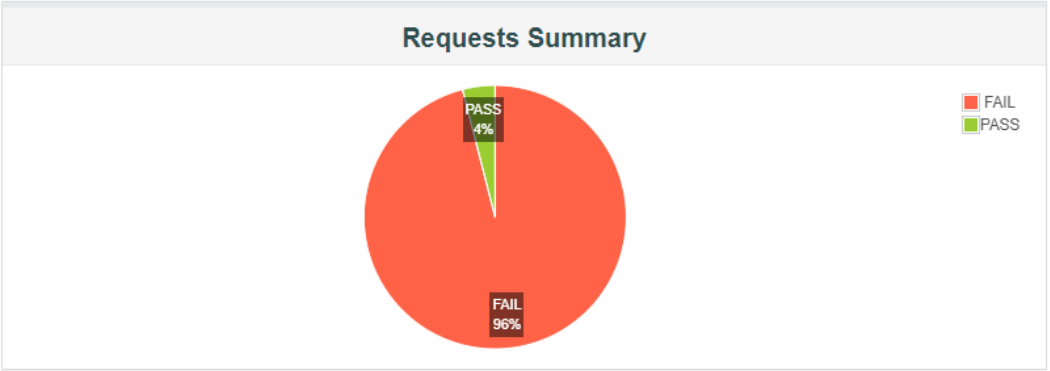
- 500 użytkowników,
- 1500 treningów,
- 250 ćwiczeń,
- 30 tysięcy statystyk odnośnie spalonych kalorii,
- 170 tysięcy statystyk odnośnie wykonanych ćwiczeń.

Wybrana baza danych spełniła oczekiwania co do pomieszczenia danych wygenerowanych przez 500 użytkowników.

Testowanie aplikacji serwerowej

Aplikacja serwerowa została przetestowana narzędziem JMeter, służącym do testowania wydajności aplikacji. Każdy test funkcji został wykonany z założeniami, które postawiono przed ich wykonaniem tj. liczbie aktywnych użytkowników = 50 oraz czasie odpowiedzi na żądanie wg tabeli.

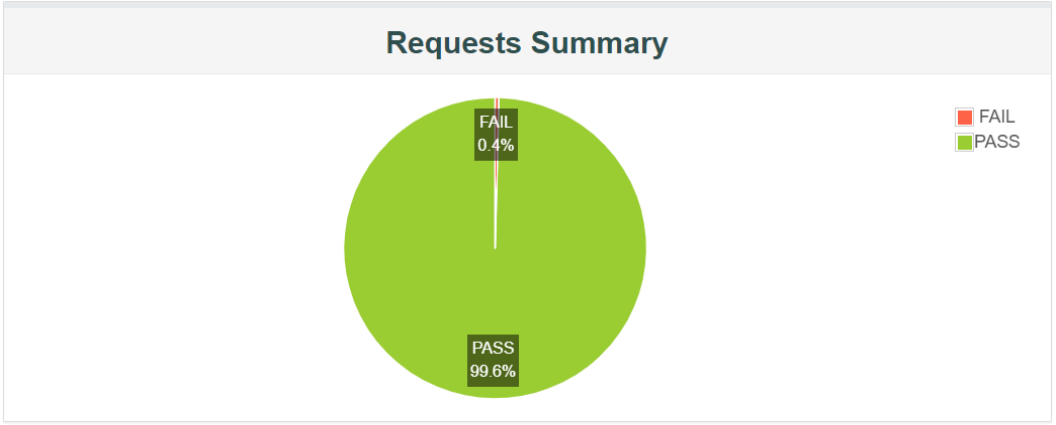
1. Tworzenie konta
   a. Raport JMeter
      i. Warunki normalne (maksymalny czas odpowiedzi: **2s**)



Wykres przedstawia zapytania, które uzyskały odpowiedzi poniżej zakładanego czasu.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	50	48	7635.54	1319	13717

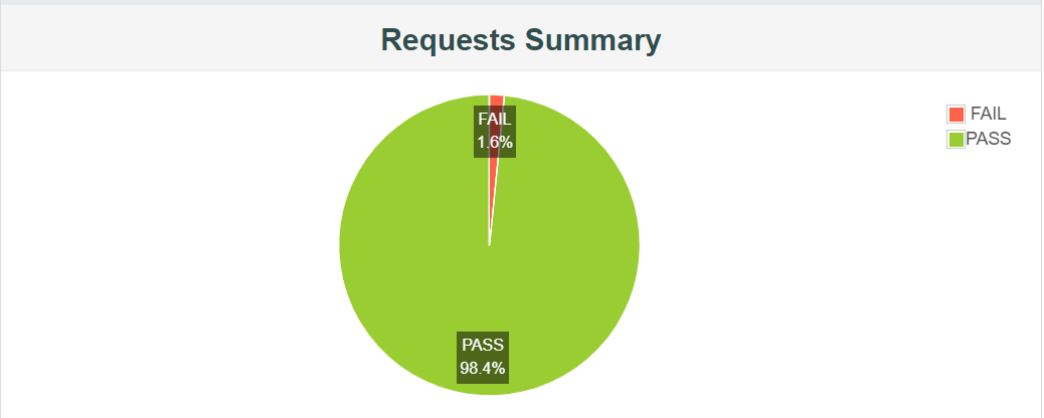
      ii. Stress test



Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	500	2	63149.16	1012	300510

iii. Crash test



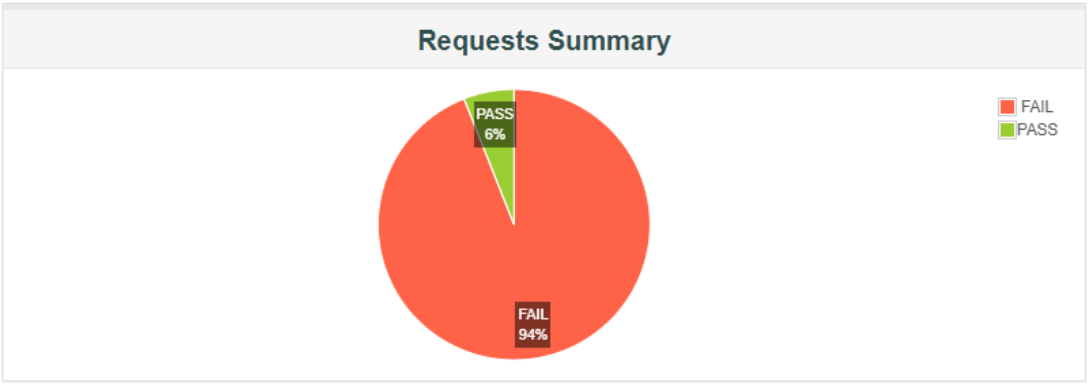
Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	1000	16	125273	1435	305713

b. Wnioski

Tylko 2 z wysłanych zapytań uzyskało odpowiedzi w założonym czasie. Średni czas odpowiedzi 3 krotnie przewyższa maksymalny czas oczekiwania. Tworzenie konta to złożona operacja a tym samym przy względnie dużym obciążeniu serwera czasy odpowiedzi znacznie się wydłużają. Przy liczbie aktywnych użytkowników równej 10, czasy odpowiedzi znacznie się zmieniają. Aż 7 żądań uzyskało odpowiedź w czasie krótszym niż założony a średni czas odpowiedzi to około 2.5 sekundy. Podczas wykonywania stress testu dopiero przy 500 aktywnych użytkownikach serwer nie odpowiedział 2 użytkownikom. Czasy odpowiedzi były bardzo długie ale większość zapytań otrzymało odpowiedź. Przy 1000 aktywnych użytkowników serwer nadal odpowiadał wysokiemu procentowi zapytań lecz czasy odpowiedzi są tak długie (średni czas odpowiedzi to 125 sekund), że można uznać że nikt z użytkowników nie czekałby tyle na odpowiedź. Limit zapytań dla funkcji tworzenia konta to około 250 aktywnych użytkowników.

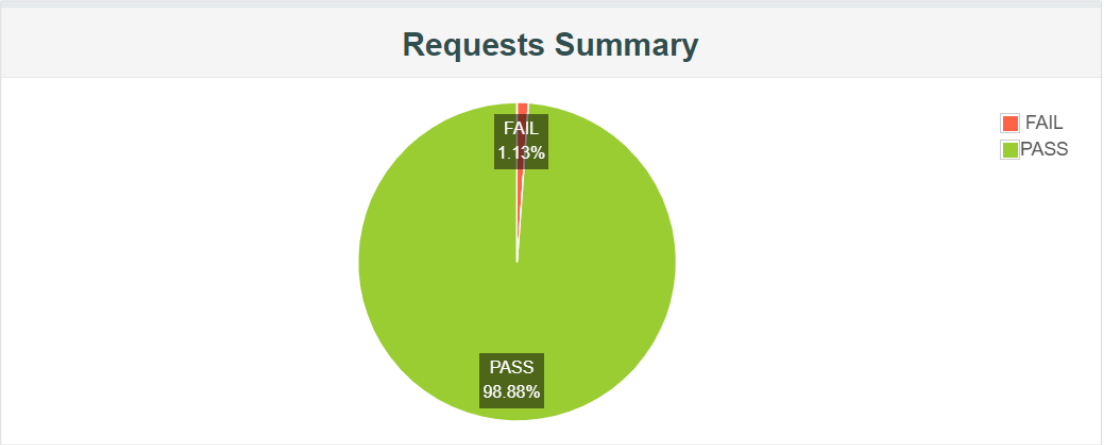
2. Logowanie na konto użytkownika.
   a. Raport JMeter
      i. Warunki normalne (maksymalny czas odpowiedzi: **2s**)



Wykres przedstawia zapytania, które uzyskały odpowiedzi poniżej zakładanego czasu.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	50	47	7367.22	1519	13264

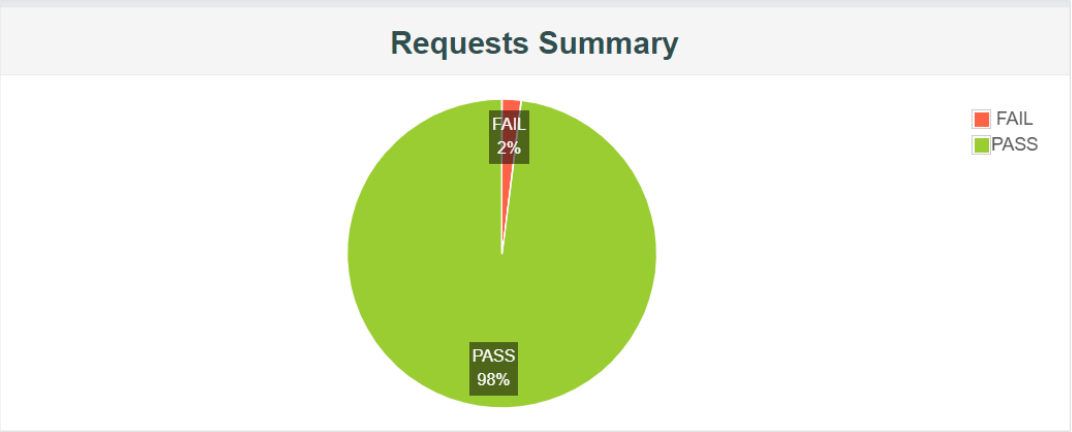
      ii. Stress test



Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	800	9	95977	1635	300577

iii. Crash test



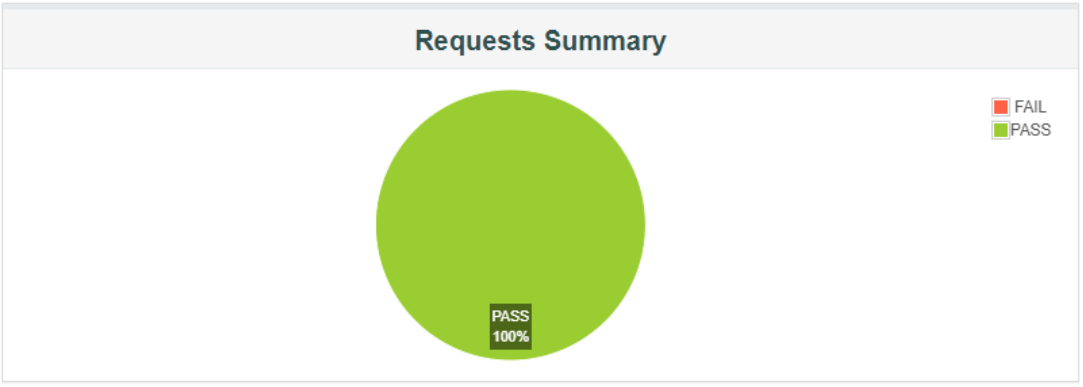
Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	1000	20	120804	1107	300534

b. Wnioski

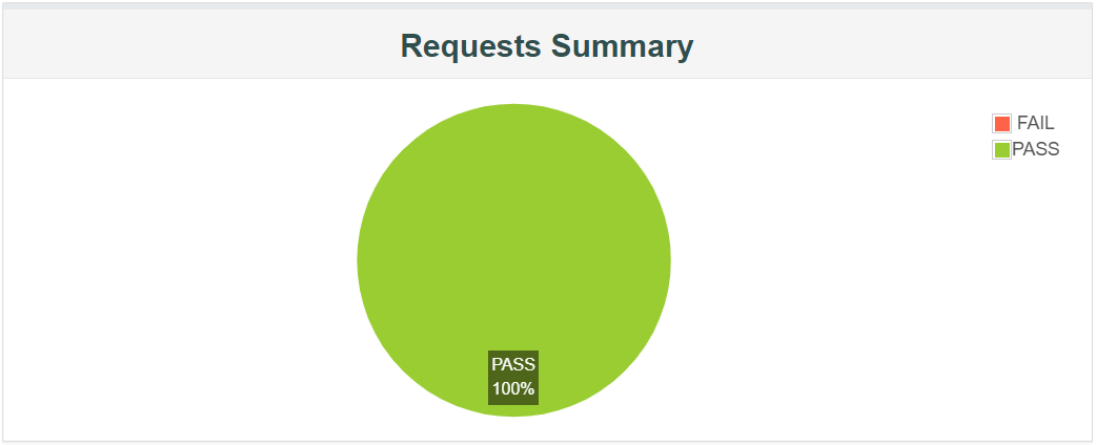
Tylko 3 spośród 50 zapytań spełniło wymaganie maksymalnego czasu odpowiedzi. Średni czas odpowiedzi był ponad 3 razy większy od zakładanego. Długie czasy odpowiedzi mogą być związane z ograniczonego planu dla deweloperów a konkretnie małej ilości RAM po stronie serwera a także winą biblioteki służącej do generowania tokenów JWT. Zauważono, że przy obciążeniu 10 użytkowników jednocześnie średnie czasy odpowiedzi wynoszą około 2.5 sekundy. Podczas wykonywania stress testu dopiero przy 800 aktywnych użytkownikach serwer nie odpowiedział 9 użytkownikom. Czasy odpowiedzi były bardzo długie ale większość zapytań otrzymało odpowiedź. Przy 1000 aktywnych użytkowników serwer nadal odpowiadał wysokiemu procentowi zapytań, lecz średni czas odpowiedzi wynosi aż 2 min. Limit zapytań dla funkcji logowania to około 250 aktywnych użytkowników.

3. Utworzenie treningu.
   a. Maksymalny czas odpowiedzi: **4s**
   b. Raport JMeter



Wykres przedstawia zapytania, które uzyskały odpowiedzi poniżej zakładanego czasu.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	50	0	1364.36	1143	1601



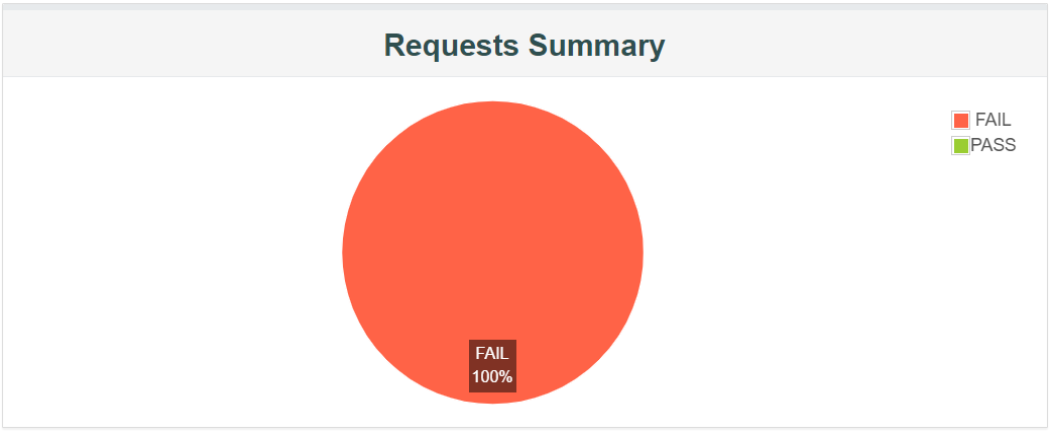
Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	500	0	5003.56	3218	6703

   c. Wnioski

   Funkcja spełnia postawione wymagania czasowe, co więcej są one lepsze od zakładanych prawie 4 krotnie. Próby testowania tej funkcji na 500 i 1000 aktywnych użytkownikach przyniosły także dobre rezultaty przez co zaprzestano dalszych testów z powodu obaw o naliczenie opłat jakie występują przy wybranym planie hostingu.

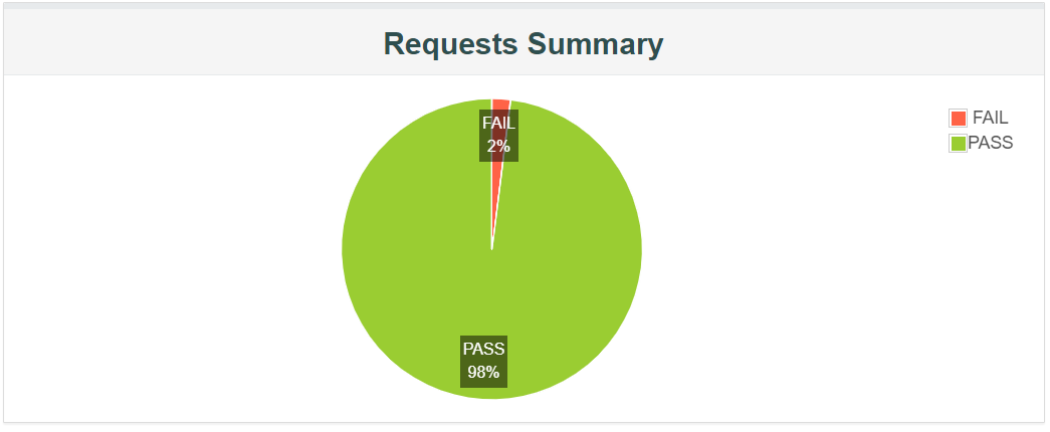
4. Wyświetlenie listy treningów.
   a. Raport JMeter
      i. Warunki normalne (maksymalny czas odpowiedzi: **2s**)



Wykres przedstawia zapytania, które uzyskały odpowiedzi poniżej zakładanego czasu.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	50	50	19372.26	2252	36200

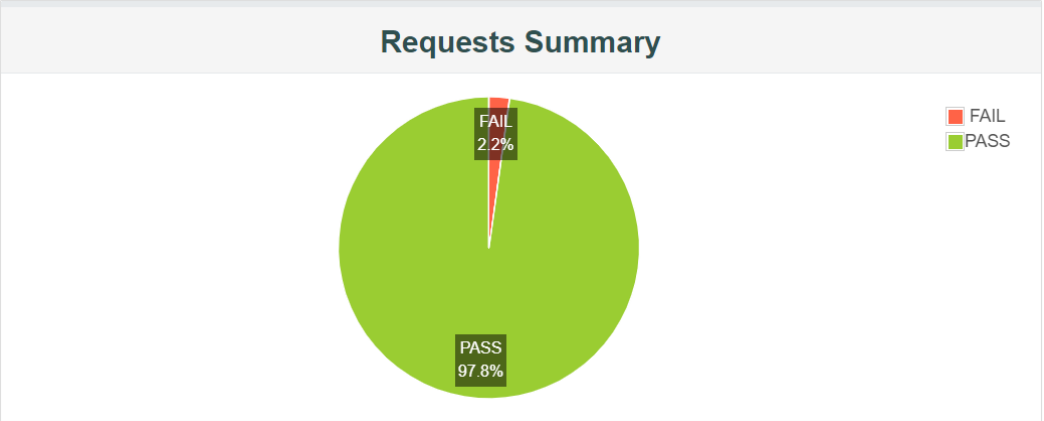
      ii. Stress test



Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	350	7	107731	2416	302584

iii. Crash test



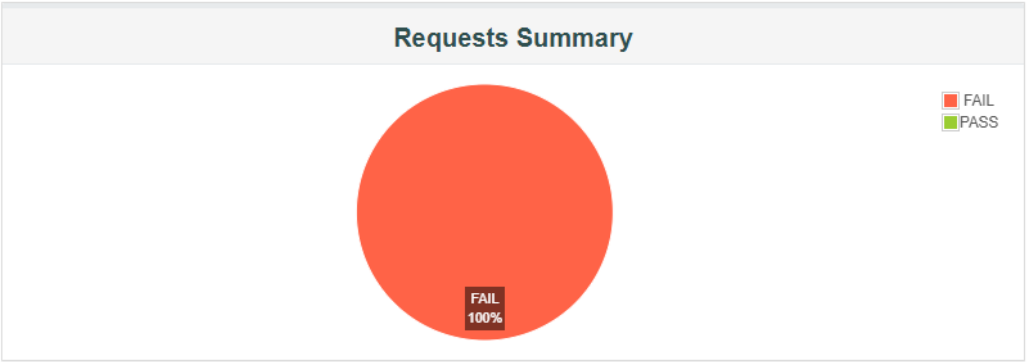
Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	500	11	149471	2641	302361

b. Wnioski

Żadne z wysłanych żądań nie spełniło wymagania maksymalnego czasu odpowiedzi. Powodem jest brak stronicowania odpowiedzi od serwera. Użytkownik wykonujący zapytanie o listę treningów, otrzymuje wszystkie dostępne treningi w bazie danych co jest wysoce kosztowne przy tak dużej ich liczbie oraz wielu zapytań wykonanych na raz. Przy zmniejszeniu liczby aktywnych użytkowników do 3 czasy zapytań wynoszą średnio 3s, co nadal nie jest satysfakcjonującym wynikiem ale zbliżonym do postawionych założeń. Stress test został przeprowadzony dla 350 aktywnych użytkowników, przy czym serwer nie odpowiedział 7 użytkownikom. Czasy odpowiedzi były bardzo długie ale większość zapytań otrzymało odpowiedź. Przy 1000 aktywnych użytkowników serwer nadal odpowiadał wysokiemu procentowi zapytań, lecz średni czas odpowiedzi wynosi prawie 150s. Limit zapytań dla tej funkcji to około 150 aktywnych użytkowników.

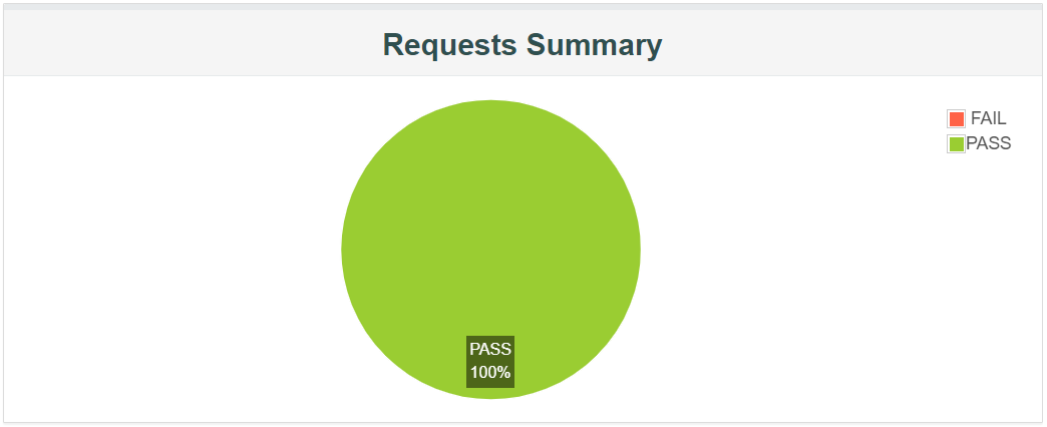
5. Przetwarzanie zakończenia treningu.
   a. Raport JMeter
      i. Warunki normalne (maksymalny czas odpowiedzi: **1s**)



Wykres przedstawia zapytania, które uzyskały odpowiedzi poniżej zakładanego czasu.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	50	50	3101.34	1190	5021

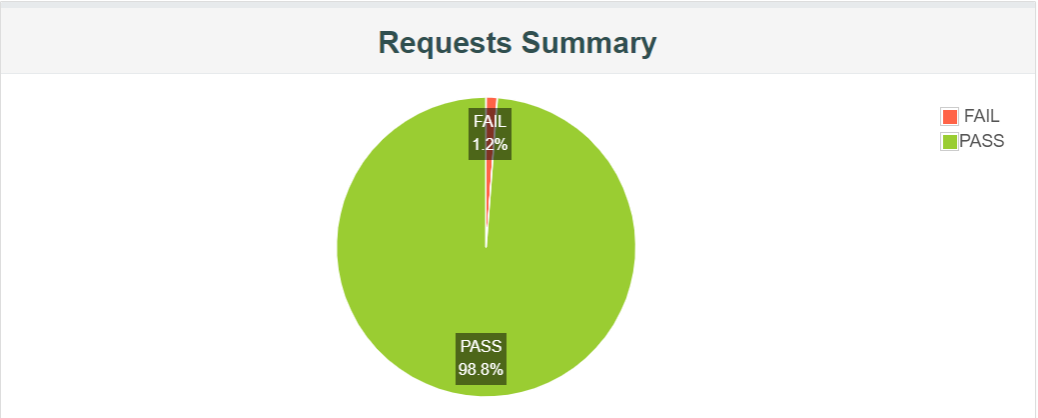
      ii. Stress test



Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	1500	0	73401.18	14961	131842

iii. Crash test



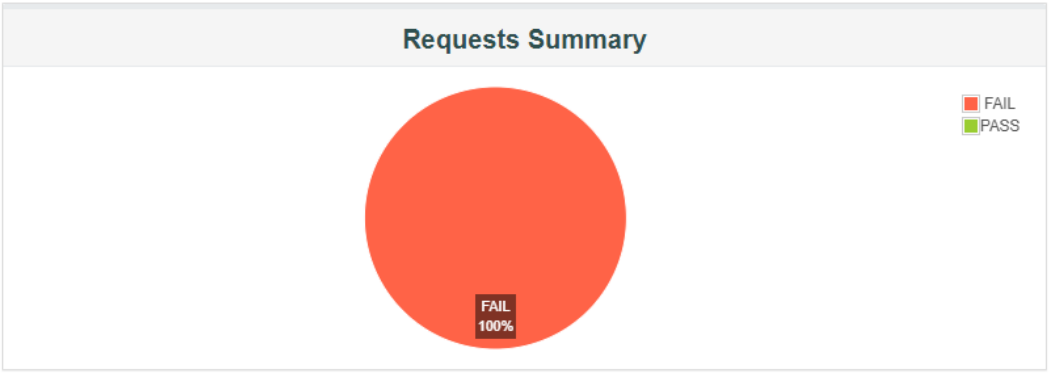
Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	2000	24	82568	2907	304786

b. Wnioski

Żadne z wysłanych żądań nie spełniło wymagania maksymalnego czasu odpowiedzi. Czasy odpowiedzi w średnim przypadku są trzykrotnie większe od zakładanych. Zauważono, że w przypadku 10 aktywnych użytkowników czasy odpowiedzi w średnim przypadku są zbliżone do postawionych i wynoszą one około 1.5s. Niepowodzenie testu wynika z niedoszacowania czasu potrzebnego do obsługi 50 użytkowników jednocześnie przy obecnej specyfikacji serwera. Stress test przy obciążeniu 1500 aktywnych użytkowników pokazał znaczne wydłużenie się czasu odpowiedzi przy zadanym obciążeniu, co ciekawe każdy z użytkowników dostał odpowiedź na swoje zapytanie. Podczas crash testu przy 2000 aktywnych użytkownikach zauważono, że aż 20 użytkowników nie dostało odpowiedzi przy czym średni czas wyniósł ponad 80s, co można uznać za brak odpowiedzi. Limit zapytań dla tej funkcji ustalono na około 350 użytkowników.

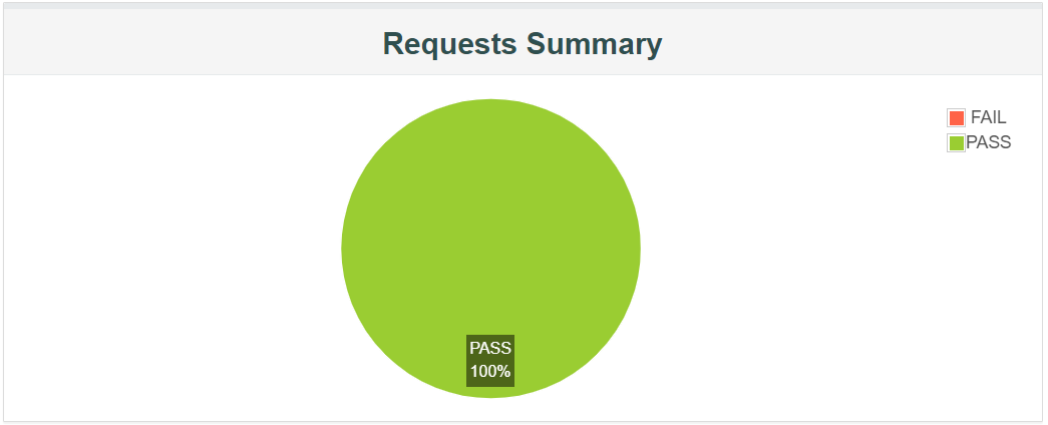
6. Wystawianie oceny treningu.
   a. Raport JMeter
      i. Warunki normalne (maksymalny czas odpowiedzi: **1s**)



Wykres przedstawia zapytania, które uzyskały odpowiedzi poniżej zakładanego czasu.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	50	50	2219.38	1975	2495

      ii. Stress test



Wykres przedstawia zapytania, które uzyskały odpowiedzi.

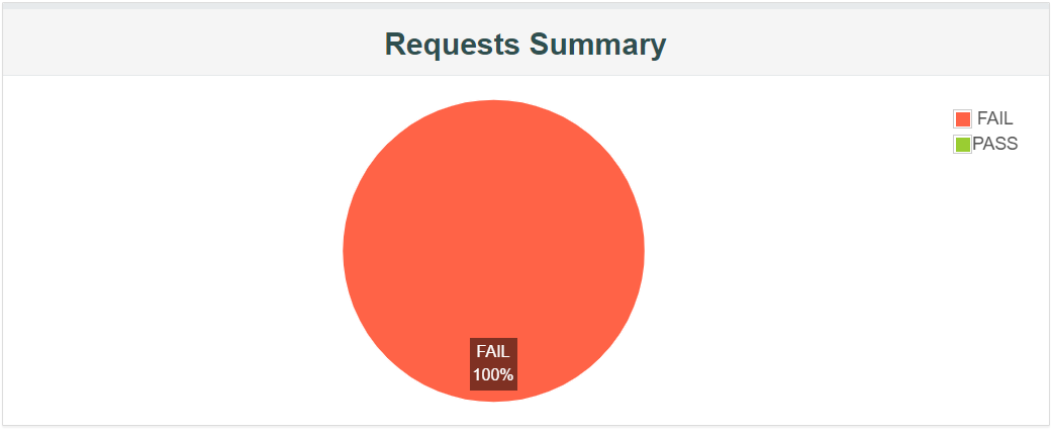
Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	2000	0	13231.66	3535	22981

   b. Wnioski

Żadne z wykonanych zapytań nie uzyskało odpowiedzi w założonym czasie. Czas w średnim przypadku wynosi 2.2s co jest bliskie oczekiwaniom. Z dalszych testów wynikło, że przy 30 aktywnych użytkownikach czas odpowiedzi wynosi około 1.2s.

Stress test wykazał wysoką odporność serwera w obszarze tej funkcji, przy 2000 aktywnych użytkowników każdy z nich dostał odpowiedź po około 13s co biorąc pod uwagę zadane obciążenie jest dobrym wynikiem. Dalsze testy nie zostały przeprowadzone z uwagi na obawy o naliczenie opłat jakie występują przy wybranym planie hostingu.

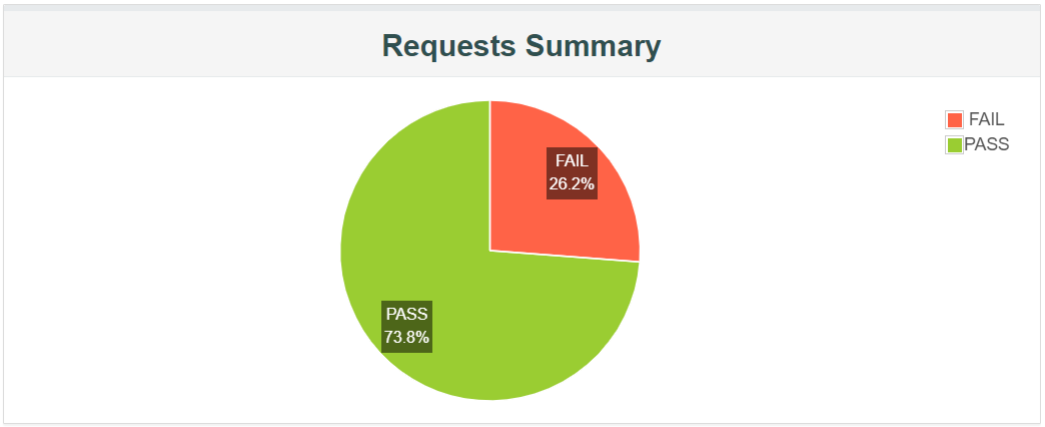
7. Czas generowania statystyk spalonych kalorii.
   a. Raport JMeter
      i. Warunki normalne (maksymalny czas odpowiedzi: **3s**)



Wykres przedstawia zapytania, które uzyskały odpowiedzi poniżej zakładanego czasu.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	50	50	3900.30	3877	3949

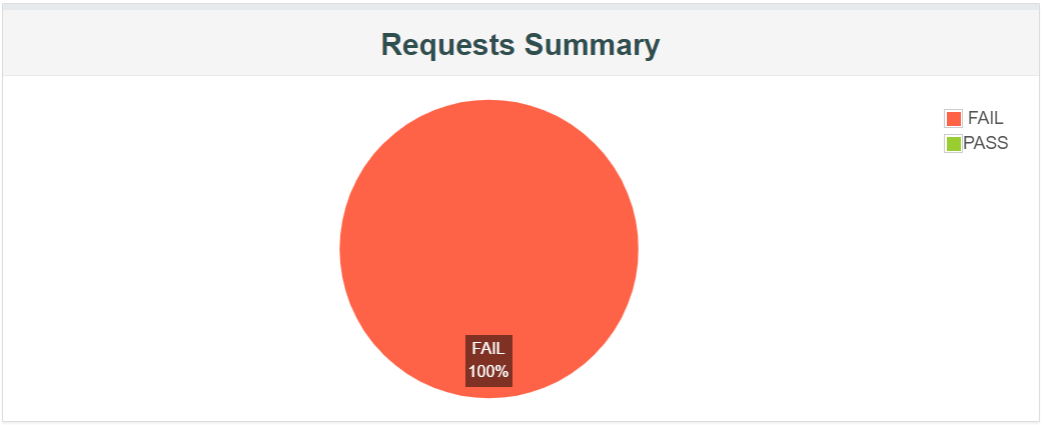
      ii. Stress test



Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	2000	524	23298.40	17315	29082

iii. Crash test



Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	3000	3000	21017.92	19042	21961

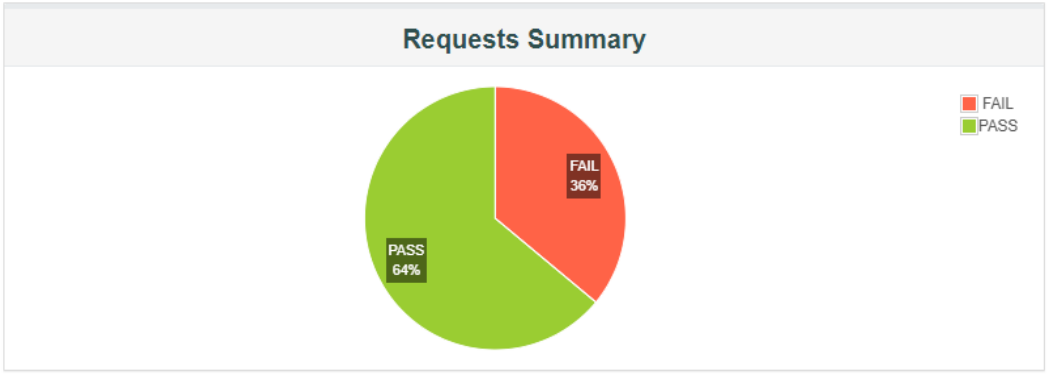
b. Wnioski

Spośród wysłanych zapytań przy warunkach normalnych, żadne nie spełniły wymagań czasowych. Podczas stress testu zauważono, że przy 2000 aktywnych użytkowników aplikacja przestaje odpowiadać około ¼ zapytań. Przy wysłaniu 3000 zapytań na raz doszło do awarii serwera.

8. Czas generowania statystyk odbytych ćwiczeń.

a. Raport JMeter

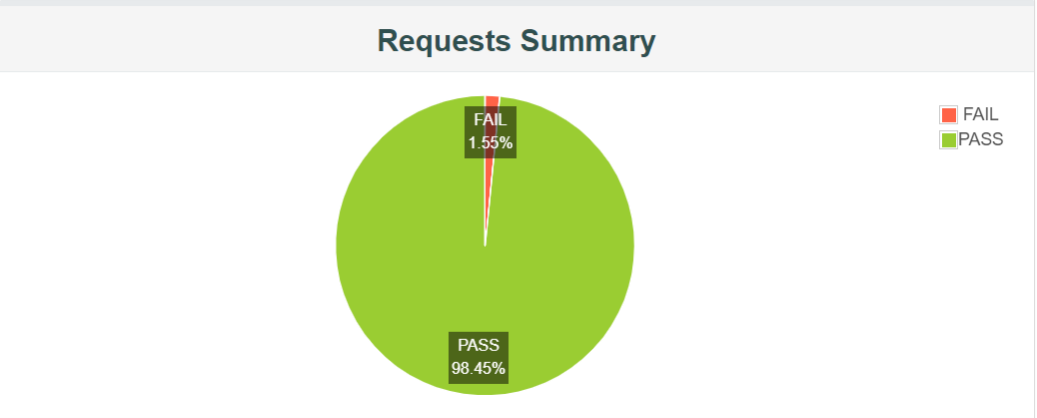
i. Warunki normalne (maksymalny czas odpowiedzi: **3s**)



Wykres przedstawia zapytania, które uzyskały odpowiedzi poniżej zakładanego czasu.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	50	18	2586.68	1197	3977

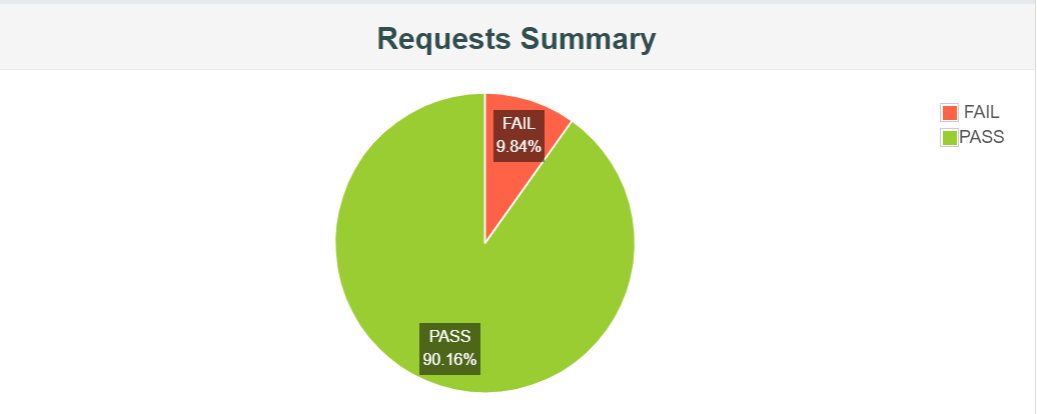
ii. Stress test



Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	2000	31	76261.59	4192	307149

iii. Crash test



Wykres przedstawia zapytania, które uzyskały odpowiedzi.

Żądania	Wykonania		Czas odpowiedzi [ms]		
	Próbki	Porażki	Średnia	Min	Max
Całkowicie	2500	246	80787.31	5055	315960

b. Wnioski
Spośród wysłanych zapytań, 32 spełniło postawione założenia czasowe. Biorąc pod uwagę średni czas odpowiedzi można uznać wyniki testu za satysfakcjonujące i spełniające postawione wymagania. Warto dodać, że przy 30 aktywnych użytkownikach, wszystkie zapytania uzyskały odpowiedź w czasie mniejszym niż 3s. Stress test wykazał, że przy 2000 jednoczesnych zapytań serwer przestaje odpowiadać. Przy 2500 aktywnych użytkownikach około 10% nie uzyskało odpowiedzi na swoje żądanie, lecz pozostałe czasy odpowiedzi były tak długie, że uznano iż nigdy nie zostałyby odebrane.

Testy integracyjne UI

Testy przeprowadzono z wykorzystaniem biblioteki WidgetTester i przeglądarki Google Chrome v112. Każdy test został uruchomiony 10-krotnie.

Tworzenie konta

Czas [s]		
Średnia	Min	Max
12.20	11.82	13.18

Wyświetlenie treningów

Czas [s]		
Średnia	Min	Max
23.15	22.89	23.65

Wyświetlenie statystyk spalonych kalorii

Czas [s]		
Średnia	Min	Max
46.12	45.81	46.78

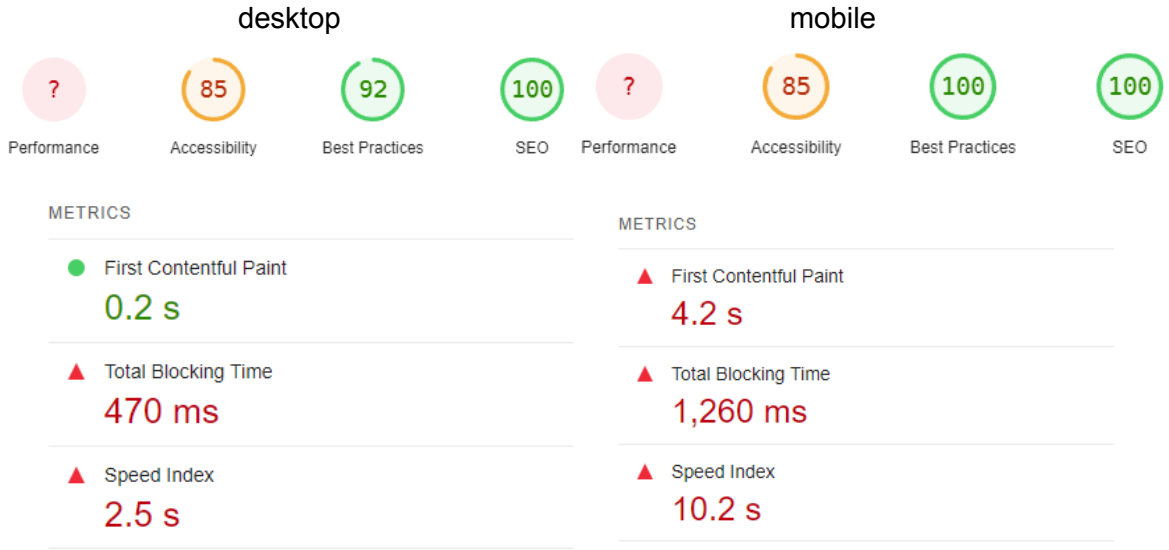
Wyświetlenie statystyk wykonanych ćwiczeń

Czas [s]		
Średnia	Min	Max
41.25	41.04	42.01

Testy wymagające załadowania obrazów trwają znacznie dłużej niż testy wymagające responsywności ze strony użytkownika (tworzenia konta).

Audyt aplikacji - Google Lighthouse

Dodatkowo wygenerowano raporty z audytu aplikacji przeprowadzonego przez wtyczkę Google Lighthouse. Raporty wygenerowano dla wersji komputerowej jak i mobilnej.



Szybsze czasy ładowania oraz krótszy czas braku możliwości podjęcia akcji w wersji desktopowej spowodowane są tym, że aplikacja została stworzona dla wersji desktopowej. Dłuższe czasy w wersji mobilnej spowodowane są niewłaściwą transformacją strony z wersji desktopowej w mobilną.

Optymalizacja

Po wykonanych testach zauważono wąskie gardła aplikacji oraz pomysły na ich usunięcie/poprawienie:

- **Wąskie gardło:** Odpowiedź otrzymywana po wykonaniu zapytania na endpoint odpowiedzialny za wyświetlenie planów treningowych nie jest stronicowana co w przypadku dużej ilości danych w bazie danych wpływa negatywnie na wydajność aplikacji serwerowej.
  Rozwiązanie: Dodanie paginacji.

- **Wąskie gardło:** Komenda uruchamiająca aplikację serwerową nie jest sparametryzowana parametrem “workers”, który określa liczbę procesów.
  Rozwiązanie: Dobranie optymalnej wartości tego parametru.

- **Pomysł:** Zaktualizowanie Fluttera do najnowszej wersji 3.10, która zwiększa wydajność aplikacji w przeglądarce internetowej.

- **Pomysł:** Aktualizacja bibliotek aplikacji internetowej do najnowszych wersji.

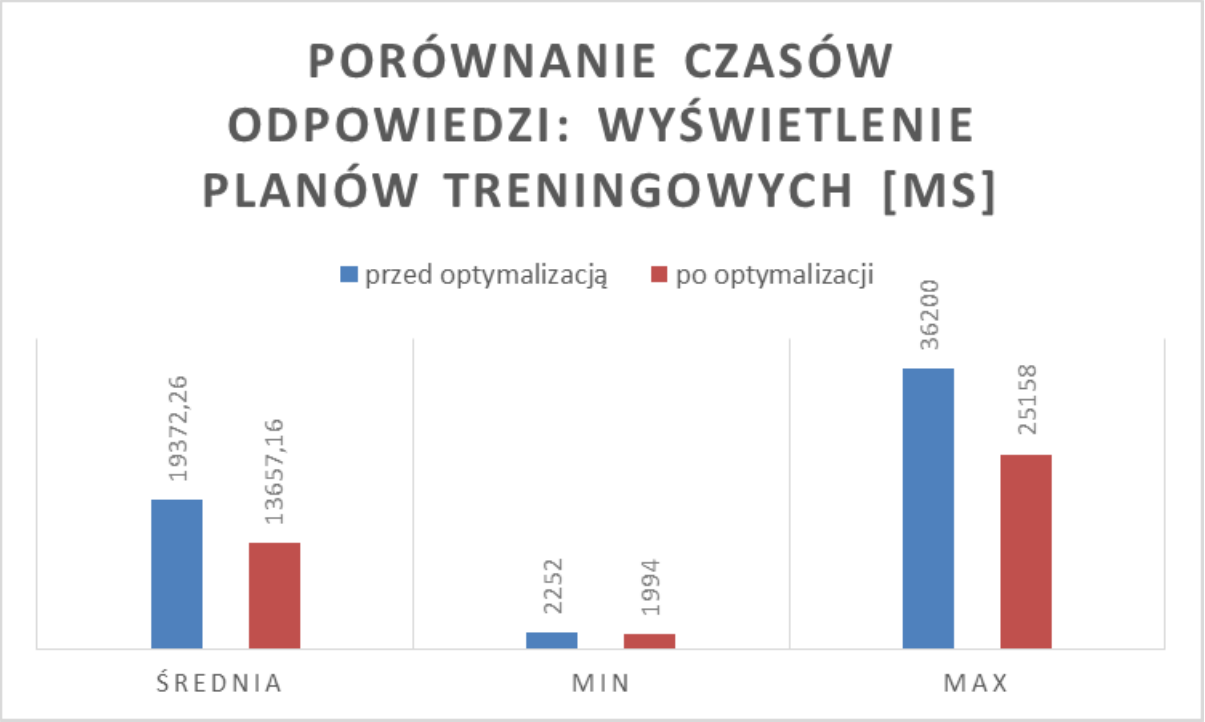
Dodanie paginacji

Zadanie opierało się na dodaniu fragmentu kodu do części aplikacji serwerowej odpowiadającej za plany treningowe.

```python
class DefaultPagination(PageNumberPagination):
    page_size = 15
```

```python
class WorkoutViewSet(viewsets.ModelViewSet):
    """
    API endpoint that allows exercises to be viewed or edited.
    """
    queryset = models.Workout.objects.all()
    serializer_class = serializers.WorkoutSerializer
    parser_classes = (MultiPartParser, FormParser)
    filter_backends = [DjangoFilterBackend, filters.SearchFilter]
    search_fields = ['title', 'description', 'author__username']
    filterset_fields = ['visibility', 'author__id']
    pagination_class = DefaultPagination
```

Rezultaty widoczne są na poniższym wykresie.



Parametryzacja komendy uruchamiającej aplikację serwerową.

Według dokumentacji Gunicorn, parametr workers powinien zostać obliczony ze wzoru:

$$(2 * liczba\_rdzeni) + 1$$

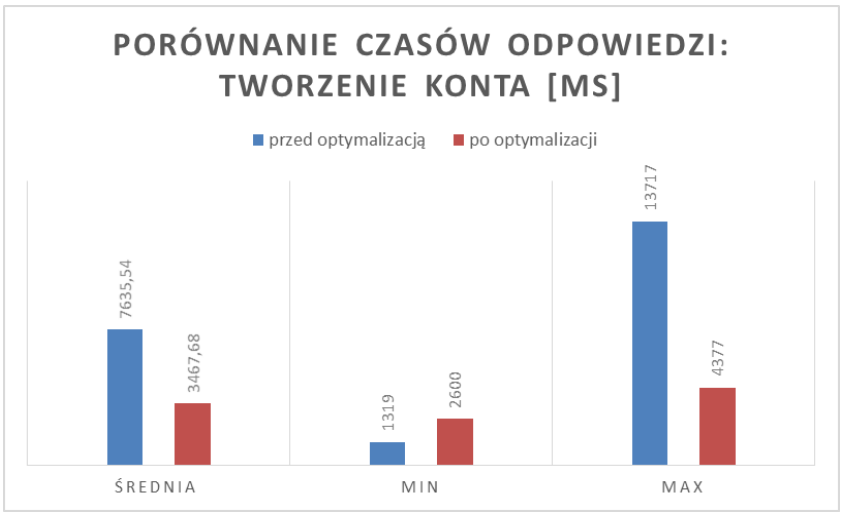
Podstawiając dane do wzoru otrzymujemy: (2 * 8) + 1 = 17.

Parametryzując komendę uzyskujemy:

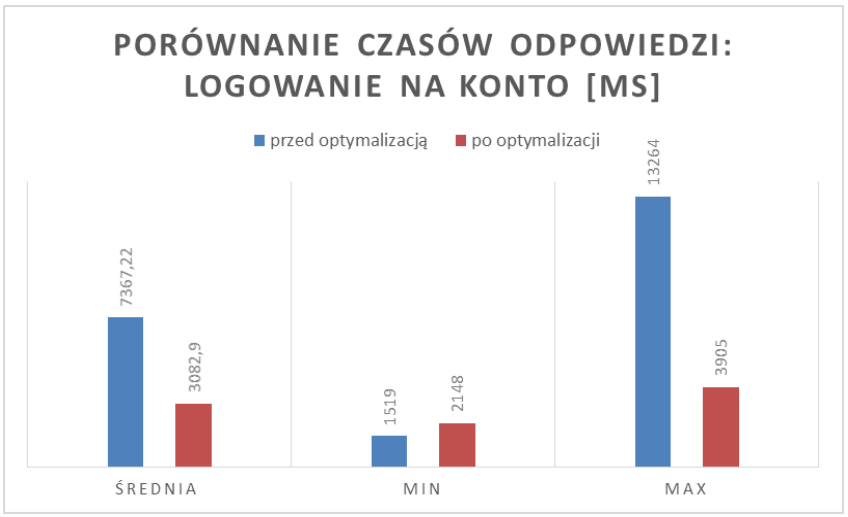
```
#!/bin/bash
python manage.py migrate & python manage.py initstats & gunicorn gymshareapi.wsgi:application -w 17
```

Wyniki optymalizacji aplikacji serwerowej (dla 50 aktywnych użytkowników):

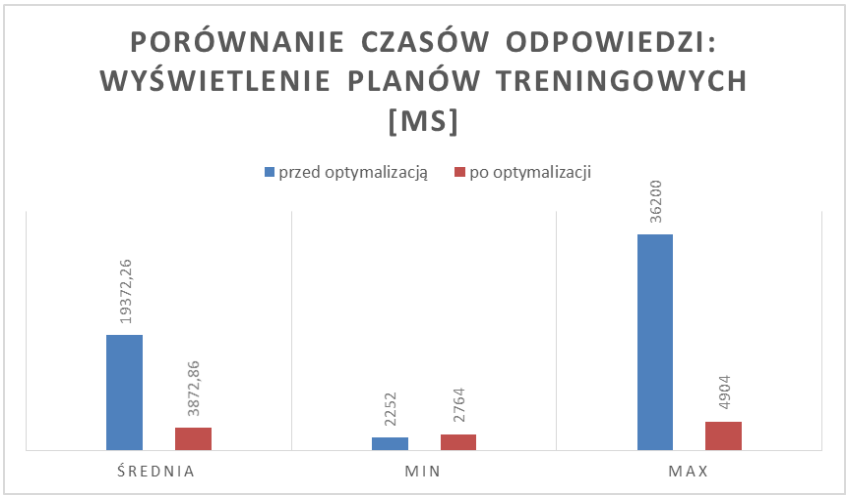
- Tworzenie konta



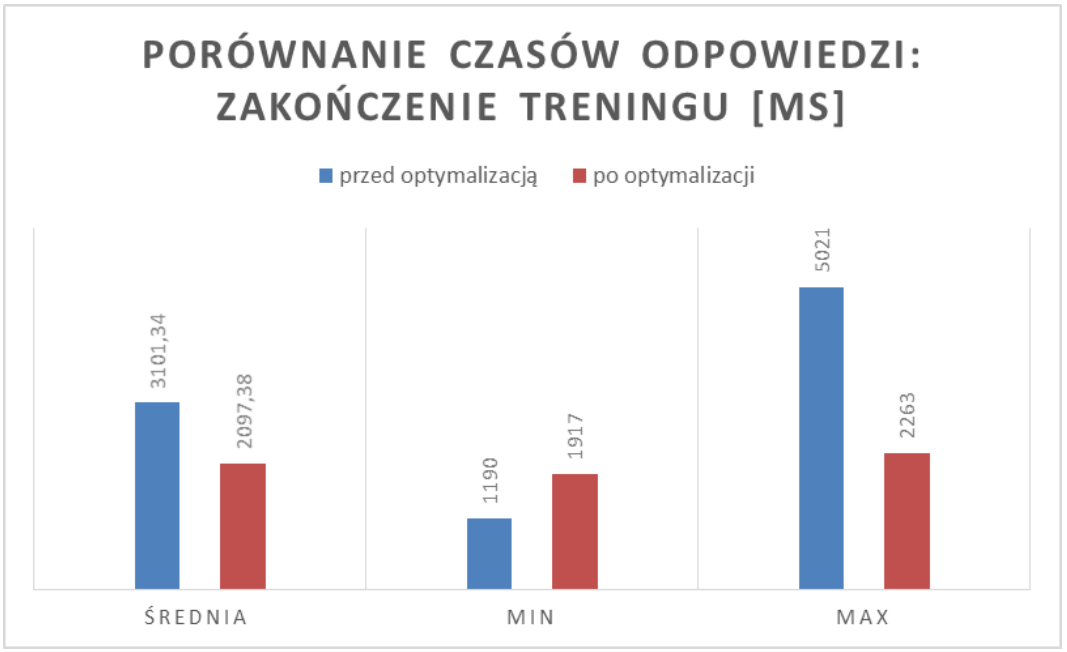
- Logowanie się do konta



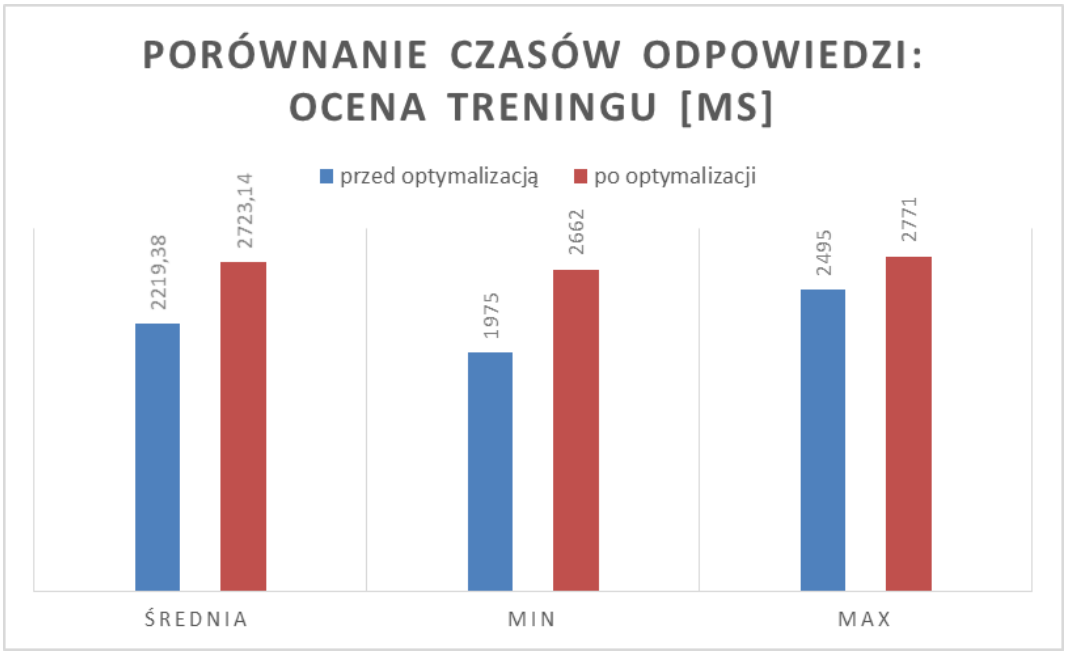
- Wyświetlanie planów treningowych



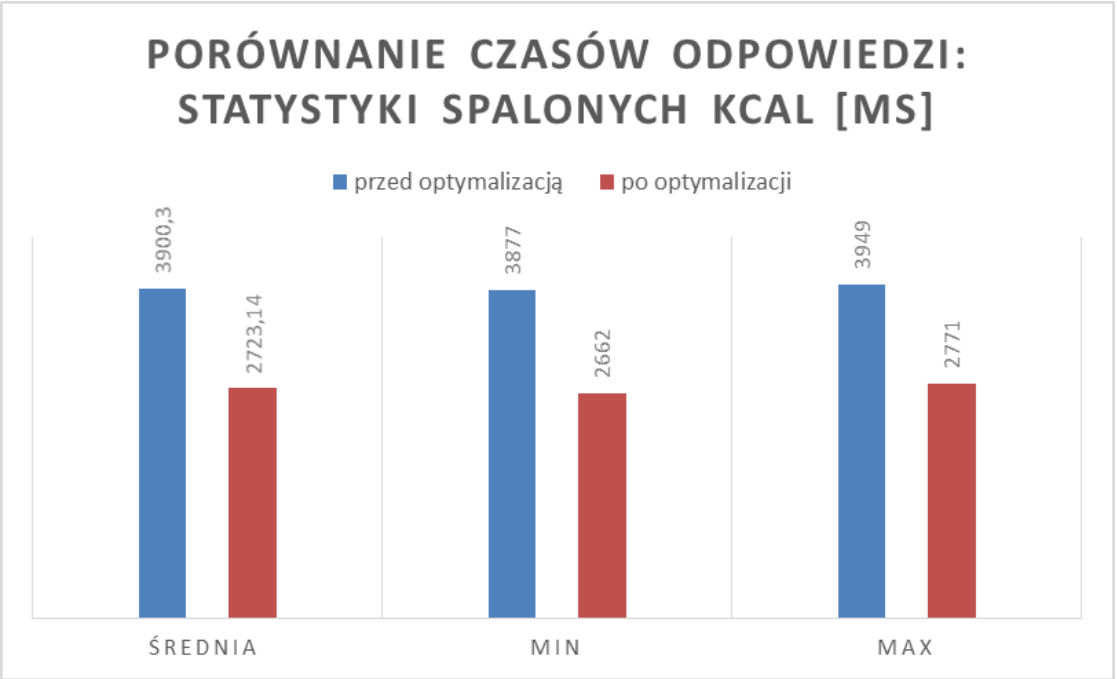
- Przetwarzanie zakończenia treningu



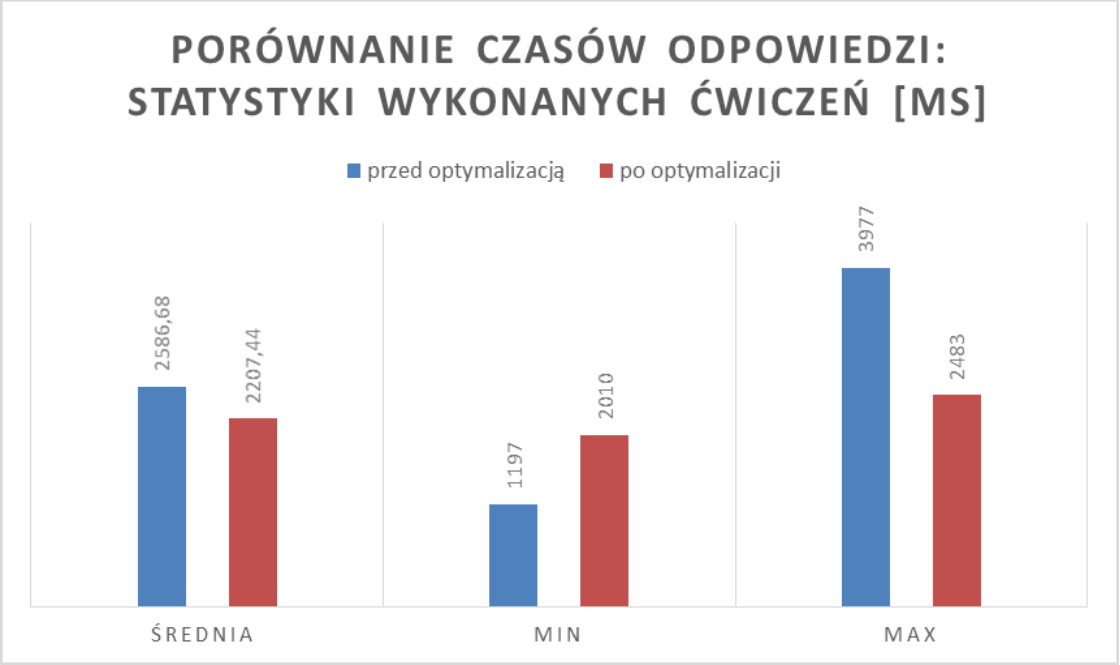
- Wystawienie oceny treningu



- Czas generowania statystyk spalonych kalorii



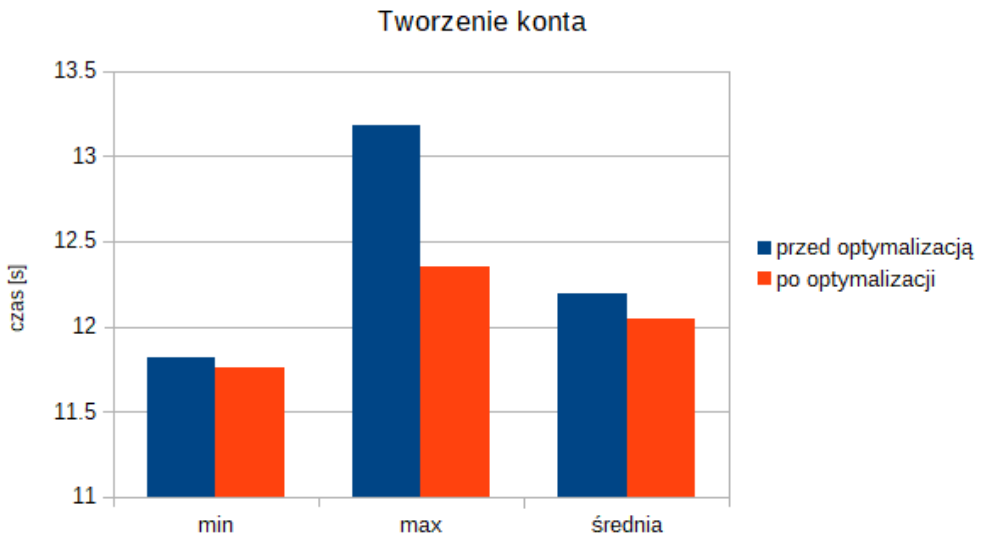
- Czas generowania statystyk odbytych ćwiczeń



Wyniki testów integracyjnych UI po optymalizacji

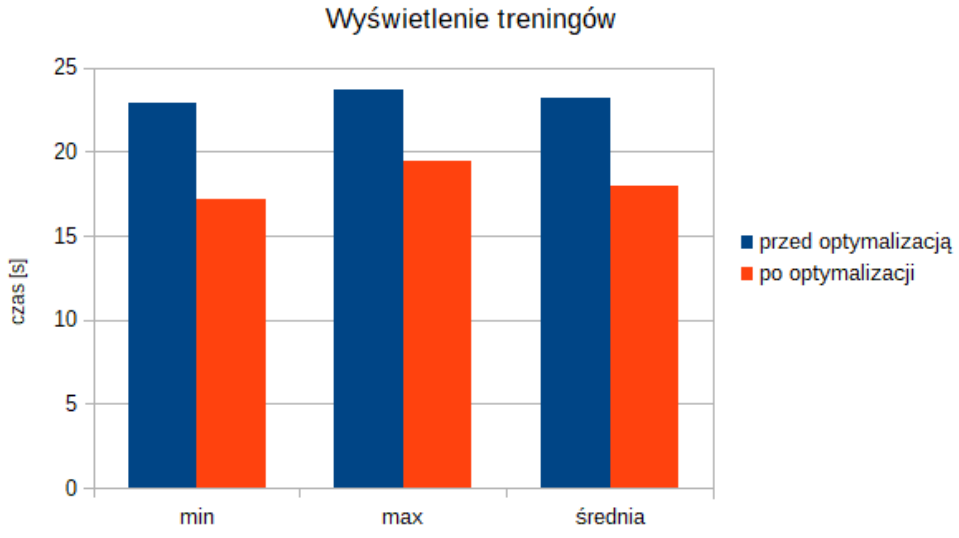
Tworzenie konta

Czas [s]		
Średnia	Min	Max
12.05	11.76	12.35



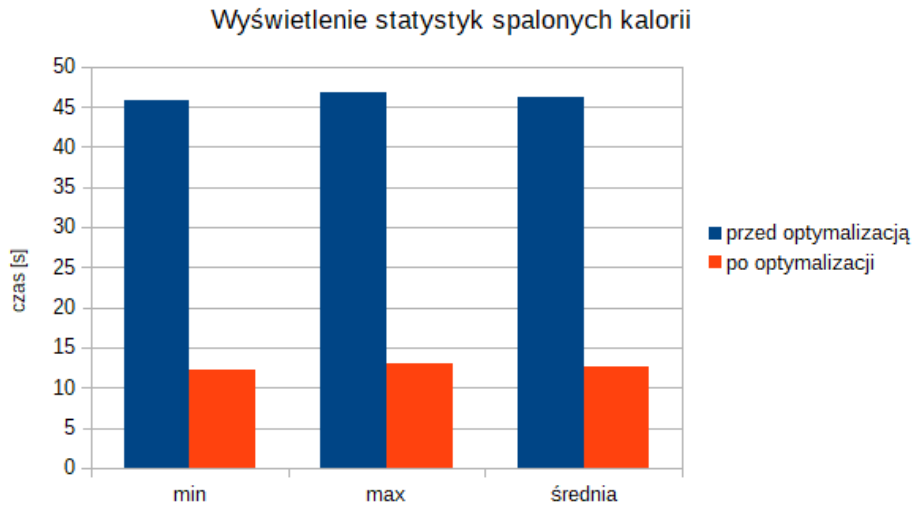
Wyświetlenie treningów

Czas [s]		
Średnia	Min	Max
17.99	17.16	19.43



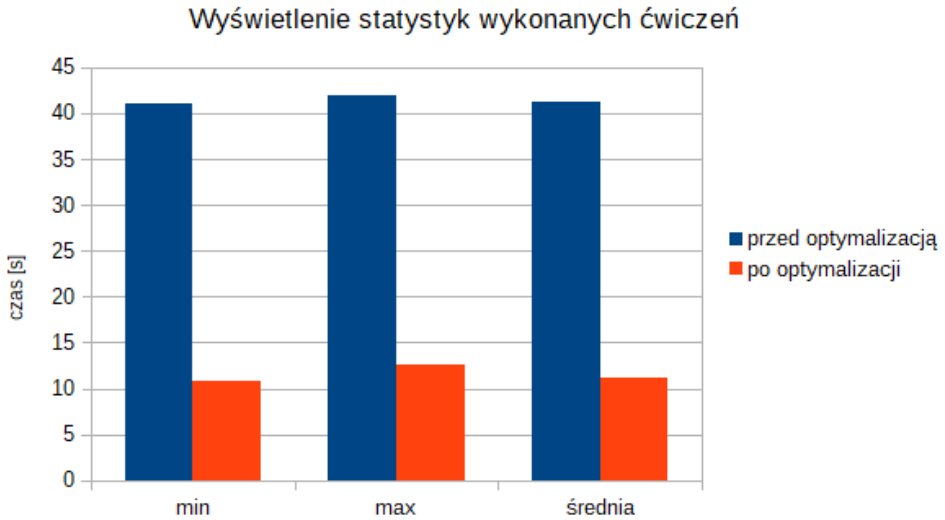
Wyświetlenie statystyk spalonych kalorii

Czas [s]		
Średnia	Min	Max
12.56	12.24	12.91



Wyświetlenie statystyk wykonanych ćwiczeń

Czas [s]		
Średnia	Min	Max
11.20	10.74	12.52



Testy wymagające załadowania obrazów trwają dłużej niż testy wymagające responsywności ze strony użytkownika (tworzenia konta).

Wyniki audytu aplikacji przy użyciu narzędzia Google Lighthouse

przed optymalizacją

desktop

METRICS

- First Contentful Paint 0.2 s
- Total Blocking Time 470 ms
- Speed Index 2.5 s

mobile

METRICS

- First Contentful Paint 4.2 s
- Total Blocking Time 1,260 ms
- Speed Index 10.2 s

po optymalizacji

desktop

METRICS

- First Contentful Paint 0.2 s
- Total Blocking Time 900 ms
- Speed Index 2.3 s

mobile

METRICS

- First Contentful Paint 0.8 s
- Total Blocking Time 1,400 ms
- Speed Index 6.2 s

Wnioski

Nazwa funkcji	Przewidywany czas odpowiedzi [s]	Średni czas odpowiedzi przed optymalizacją [s]	Średni czas odpowiedzi po optymalizacji [s]
Tworzenie konta	2	7,6	3,4
Logowanie na konto użytkownika	2	7,3	3,0
Utworzenie treningu	4	1,3	—
Wyświetlenie listy treningów	2	19,3	3,8
Przetwarzanie zakończenia treningu	1	3,1	2,0
Wystawienie oceny treningu	1	2,2	2,7
Czas generowania statystyk spalonych kalorii	3	3,9	2,7
Czas generowania statystyk odbytych ćwiczeń	3	2,5	2,2

1. **Dodanie paginacji.**
   Wyniki (przy 50 aktywnych użytkownikach) uległy znacznej poprawie. Średni czas oczekiwania spadł o prawie 6s, minimalny czas odpowiedzi nieznacznie się skrócił a maksymalny czas odpowiedzi został skrócony o ponad 11s. Usunięcie wąskiego gardła nie spowodowało, że wyniki są zgodne z założeniami postawionymi przed wykonaniem testów, lecz widać ogromną poprawę.

2. **Parametryzacja komendy uruchamiającej aplikację serwerową.**
   Sparametryzowanie komendy uruchamiającej aplikację serwerową spowodowało obniżenie średnich, minimalnych i maksymalnych czasów odpowiedzi na zapytania dla wszystkich funkcjonalności. Zmiana została uznana za korzystną, ponieważ sprawiła znaczną poprawę wydajności aplikacji serwerowej i pozwoliła się zbliżyć do oczekiwanych założeń, postawionych przed etapem testów.

3. **Zaktualizowanie Fluttera i pluginów do najnowszej wersji.**
   Najnowsza wersja Fluttera (3.10) usprawnia wczytywanie obrazów z API, co można zauważyć po kilkukrotnej poprawie czasów wykonania testów integracyjnych wymagających ładowania obrazów.
   Optymalizacja poprawiła czasy FCP i Speed Index dla wersji mobilnej (aktualizacja bibliotek i Fluttera poprawiła transformację strony z wersji desktopowej w mobilną). W obydwu przypadkach, zwiększył się czas Total Blocking Time w związku z większą ilością Long Tasków (tasków wykonujących się dłużej niż 50 ms).

4. **Optymalizacja komercyjna.**

Przy obciążeniu 50 użytkowników potrzeba 0.45 CPU i 880 MB RAMu. Takie same czasy dla 100 użytkowników można osiągnąć przy zwiększeniu zasobów do ok. 1 CPU oraz prawie 2 GB RAMu. Takie same czasy dla 200 użytkowników wymagać będą 2 CPU oraz 4 GB RAMu. Stosunek liczby użytkowników do przyrostu zasobów przypomina funkcję liniową.

Na podstawie wykonanych testów oszacowano, że wraz z wzrostem liczby użytkowników, czasy odpowiedzi będą rosnąć liniowo a po przekroczeniu około 250 użytkowników zaobserwowano by wzrost wykładniczy (przy aktualnej specyfikacji serwera). Odpowiedzią mogłoby być zwiększenie zasobów serwera według powyższych instrukcji.