JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 31 (2024) 132–137
Received: 25 April 2024
Accepted: 21 May 2024

Comparison of Laravel and Symfony - the most popular PHP frameworks, based on a simple CRUD application

Porównanie Laravel i Symfony - najpopularniejszych szkieletów programistycznych języka PHP, na podstawie prostej aplikacji typu CRUD

Patryk Gajewski*, Mariusz Dzieńkowski

Department of Computer Science, Lublin University of Technology, Nadbystrzycka 36B, 20-618 Lublin, Poland

Abstract
The main purpose of the study is to compare the performance of two popular PHP language programming frameworks, Laravel 10 and Symfony 6, based on a CRUD test applications. The benchmarking also took into account the specifics of the implementation process and the size of the developer community support. The study was carried out using two simple and small test applications with the same set of functionalities. Performance was evaluated on the basis of execution times for single CRUD operations and simultaneous handling of 10, 100, 500 and 1,000 requests. Execution times - from the moment the request is sent to the server, to the moment the response is received - were measured using the Postman tool. The results showed that Symfony 6 is a better solution for applications where performance is crucial.

Keywords: PHP frameworks; performance analysis; Symfony; Laravel

Streszczenie
Głównym celem pracy było porównanie wydajności dwóch popularnych szkieletów programistycznych języka PHP – Laravel 10 oraz Symfony 6 na podstawie aplikacji testowej typu CRUD. W ramach analizy porównawczej uwzględniono również specyfikę procesów implementacji oraz wielkości wsparcia społeczności programistów. Badanie zostało przeprowadzone przy użyciu dwóch prostych i niewielkich aplikacji testowych, posiadających ten sam zestaw funkcjonalności. Wydajność została oceniona na postawie czasów wykonywania pojedynczych operacji CRUD oraz jednoczesnej obsługi 10, 100, 500 i 1000 żądań. Pomiary czasów wykonania operacji - od momentu wysłania żądania do serwera, do momentu otrzymania odpowiedzi, były dokonywane za pomocą narzędzia Postman. Wyniki pokazały, że Symfony 6 jest lepszym rozwiązaniem dla aplikacji, gdzie wydajność ma zasadnicze znaczenie.

Słowa kluczowe: szkielet programistyczny języka PHP; analiza wydajnościowa; Laravel; Symfony

*Corresponding author
Email address: patryk.gajewski@pollub.edu.pl (P. Gajewski)

1. Wstęp

We współczesnym oraz dynamicznie rozwijającym się środowisku online z postępującym procesem cyfryzacji oraz przenoszeniem tradycyjnej sprzedaży na platformy internetowe aplikacje webowe zyskują coraz bardziej na popularności. Szkielety programistyczne stanowią kluczowy element w budowie aplikacji, ze względu na dostarczanie gotowych zestawów funkcjonalności [1] oraz modułów, które przyśpieszają znacznie proces implementacji. Laravel oraz Symfony, jako popularne narzędzia języka PHP [2], pełnią istotną rolę w tworzeniu aplikacji. Ich wybór ma bezpośredni wpływ na wydajność, szybkość działania witryny, jakość utworzonej aplikacji oraz efektywność działania systemu.

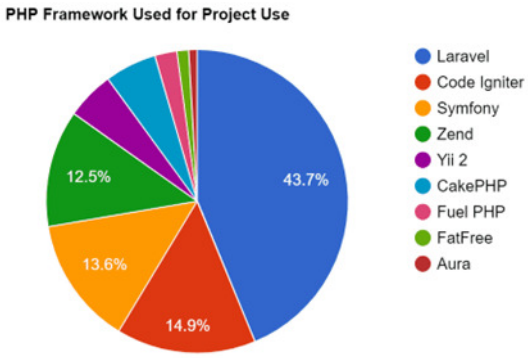
Laravel oraz Symfony, będące reprezentantami nowoczesnych podejść do projektowania aplikacji webowych, umożliwiają programistom skoncentrowanie się na implementacji konkretnych funkcji, zamiast poświęcania czasu na ponowne tworzenie podstawowych elementów systemu od zera. Wybór pomiędzy tymi dwoma szkieletami programistycznymi staje się kwestią istotną dla efektywnej pracy programisty. Optymalny wybór zależy od specyfiki projektu, jego rozmiaru, a także oczekiwań co do wydajności i skalowalności. W ten sposób, decyzja dotycząca wyboru szkieletu programistycznego staje się ważnym elementem procesu projektowania aplikacji webowej. W projektach długoterminowych oraz złożonych szkielet programistyczny Symfony wykorzystywany jest najczęściej [3]. Jego struktura składa się z systemu pakietów, z których każdy z nich realizuje określone zadanie. Laravel charakteryzuje się szybkim tworzeniem aplikacji [4], ze względu na dużą intuicyjność pisania kodu oraz wysoki poziom abstrakcji. Używany jest najczęściej do małych oraz średnich projektów, lecz również jest odpowiedni w przypadku tworzenia dużych i skomplikowanych aplikacji internetowych.

Klienci oraz użytkownicy mają coraz większe wymagania wobec serwisów, z których korzystają na co dzień. Oczekują oni, aby serwis działał płynie i szybko. Długie czasy ładowania się stron czy opóźnienia w działaniu funkcji mogą negatywnie wpływać na doświadczenie użytkownika. Ochrona prywatności i bezpieczeństwo danych osobowych są priorytetem, więc dane muszą być odpowiednio chronione oraz zabezpieczone przed wyciekami czy nieautoryzowanym dostępem. Ważnym aspektem aplikacji internetowych jest także ich skalowalność oraz elastyczność. Serwisy muszą być przygotowane na wzrost liczby użytkowników oraz zmiany

w wymaganiach biznesowych. Ponadto, interfejs użytkownika powinien być intuicyjny i łatwy w obsłudze, co ma istotne znaczenie dla pozytywnego doświadczenia użytkownika. Wspieranie różnych przeglądarek, systemów operacyjnych, urządzeń, takich jak smartfony, tablety czy komputery stacjonarne, jest nie bez znaczenia dla zapewnienia dostępności aplikacji dla różnych grup odbiorców.

Autorzy pracy skupili się na analizie porównawczej wydajności dwóch popularnych szkieletów programistycznych języka PHP: Laravel w wersji 10 i Symfony w wersji 6 na podstawie identycznej aplikacji testowej przygotowanej jako obiekt badawczy. W ramach niniejszej pracy porównano również złożoność procesu implementacji oraz wsparcie społeczności zgromadzonej wokół jednej i drugiej technologii. Celem było zrozumienie, jak oba szkielety programistyczne radzą sobie z jednoczesnym wykonywaniem pojedynczych operacji jak i setek takich samych, powtarzalnych operacji, aby poszerzyć wiedzę, która może okazać się pomocna podczas podejmowania decyzji co do doboru narzędzia do specyfiki danego projektu programistycznego.

Z rysunku 1 widać, że najczęściej wykorzystywanym szkieletem programistycznym był w 2023 Laravel (43,7%), natomiast Symfony zajmowało 3 pozycję z udziałem w rynku 14,9%.



Rysunek 1: Najczęściej używane szkielety programistyczne języka PHP w roku 2023 [5].

2. Przegląd literatury

W ostatnich latach można zaobserwować wzrostowy trend wytwarzania oprogramowania webowego. Konsekwencją tej tendencji było powstanie wielu narzędzi deweloperskich służących do tego celu: języków, bibliotek, szkieletów programistycznych. To powoduje, że programiści przed przystąpieniem do implementacji muszą dokonać wyboru odpowiedniego narzędzia, a w przypadku aplikacji www, optymalnego dla realizowanego projektu frameworka. Wielu autorów publikacji zaczęło badać wydajność [6], skalowalność i dostępność poszczególnych szkieletów programistycznych [1], aby w jak najprostszy sposób określić, który z nich będzie najlepszy do danego zadania. Na przestrzeni lat powstało wiele artykułów naukowych, gdzie można odnaleźć bardzo dużo przydatnych wniosków wypływających z przeprowadzonych analiz badawczych. W przypadku języka PHP najczęściej porównuje się dwa szkielety programistyczne Laravel oraz Symfony ze względu na ich dużą popularność, bogate możliwości oraz aktywne społeczności wspierające rozwój aplikacji opartych na tych frameworkach.

Autorzy artykułu naukowego [7] porównali trzy popularne szkielety programistyczne języka PHP: Laravel, CakePHP oraz CodeIgniter. Ich eksperyment skupiał się na analizie wydajności aplikacji podczas operacji CRUD oraz procesu wgrywania zdjęć, przy czym dodatkowo uwzględniono aspekt ponownego wykorzystania tworzonego kodu. Laravel okazał się narzędziem wydajnym, cechującym się szybszym tempem rozwoju aplikacji w porównaniu do konkurencyjnych rozwiązań. Niemniej jednak to CodeIgniter osiągnął najwyższą wydajność, charakteryzując się najkrótszymi czasami wykonania 4 typów operacji. Może to być spowodowane mniejszą liczbą plików składającą się na projekt, w porównaniu z innymi analizowanymi technologiami, co automatycznie przekłada się na efektywność działania systemu. Natomiast CakePHP wykazał się przeciętnymi wynikami wydajnościowymi w porównaniu do pozostałych dwóch szkieletów choć wypadał najlepiej pod względem możliwości ponownego wykorzystania kodu.

W badaniu opisanym w publikacji [1] autorzy przeprowadzili badanie analizy porównawczej dwóch popularnych szkieletów programistycznych języka PHP, Symfony i Laravel. Analiza opierała się na dobranych kryteriach, takich jak organizacja kodu, popularność, architektura, wymagania systemowe czy dokumentacja. Podczas przeprowadzonych badań autorzy tej publikacji doszli do wniosku, iż Symfony może być lepszym szkieletem programistycznym dla większych projektów ze względu na swoją stabilność. Laravel zaś jest łatwiejszy w nauce oraz cieszy się dużo większą popularnością wśród społeczności.

Autorzy pracy [2] opisują wyniki porównania szkieletu programistycznego Laravel z innymi technologiami, takimi jak Symfony oraz Zend, pod kątem możliwości tworzenia aplikacji webowych oraz wydajności. Przeprowadzono analizę wydajności podczas pracy z bazą danych oraz oceniono różnice w możliwościach oferowanych przez te narzędzia. Podczas przeprowadzonych badań zauważono, iż szkielet programistyczny Laravel wypadł najgorzej wydajnościowo ze wszystkich innych badanych szkieletów w przypadku rozbudowanych aplikacji internetowych. Czasy odpowiedzi serwera były zazwyczaj od 50 do 200 milisekund dłuższe, co przekładało się na wolniejsze działanie witryny internetowej, a zarazem wpływało negatywnie na komfort użytkowania aplikacji.

Autorzy artykułu naukowego [6] badali, który z analizowanych szkieletów: Laravel, Symfony czy CodeIgniter jest lepszym rozwiązaniem dla programistów wykonujących projekty stron internetowych. Wydajność została przetestowana na podstawie kilku metryk: liczby obsłużonych operacji na sekundę, zużycia pamięci, czasu odpowiedzi serwera czy liczby wymaganych plików do wykonania żądania. Dzięki zgromadzonym danym autorzy publikacji określili, który z frameworków jest najbardziej efektywny. W analizie stwierdzono, że Laravel osiąga najwyższe rezultaty w obszarze obsługi żądań na

sekundę, zdolny jest obsłużyć około 3000 z nich, co stanowi niemal dwukrotnie lepszy wynik niż w przypadku CodeIgniter oraz trzykrotnie lepszy niż w przypadku Symfony. Ponadto szkielet ten, wymaga znacznie mniejszej ilości pamięci do przetworzenia operacji, a także charakteryzuje się najkrótszym czasem odpowiedzi serwera. W kontekście metryki dotyczącej liczby plików niezbędnych do przetworzenia operacji Symfony okazał się bezkonkurencyjny. Wymaga on około 40% mniej plików w porównaniu z Laravelem oraz około 30% mniej plików niż CodeIgniter.

W artykułach naukowych autorzy publikacji porównują popularne szkielety programistyczne języka PHP, takie jak Symfony, Laravel, CakePHP oraz CodeIgniter. Podczas tych analiz wykorzystywali różne kryteria, takie jak wydajność, skalowalność czy wsparcie społeczeństwa. Badania ukazały słabe oraz silne strony każdego z tych frameworków, takie jak stabilność Symfony, szybkość rozwoju Laravela czy efektywność działania CodeIgnitera. Autorzy przeanalizowali również różnice w wydajności, wielokrotnym użyciu kodu oraz ilości potrzebnych plików do przetworzenia operacji. Pozwoliło to określić, który szkielet programistyczny może być najbardziej efektywny dla konkretnego projektu. Poniższa praca jest niejako rozszerzeniem opisanych wcześniej publikacji. Na podstawie prostej aplikacji typu CRUD, zbadano wydajność szkieletu programistycznego Laravel 10 oraz Symfony 6. Badanie wydajności opierało się na operacjach dodawania, usuwania, edytowania, wyświetlania postów blogowych oraz symulacji zwiększonego ruchu na platformie poprzez jednoczesne wykonywanie dziesiątek oraz setek takich samych operacji.

3. Metodyka badawcza

Badanie wykonane zostało przy użyciu dwóch aplikacji bazodanowych, które posiadają taki sam zestaw funkcjonalności. Obie aplikacje uruchomiono przy użyciu tego samego serwera oraz użyto takiej samej bazy do agregowania wszystkich danych. Pierwsza z nich opiera się na szkielecie programistycznym Laravel w wersji 10. Druga aplikacja oparta jest na szkielecie programistycznym Symfony w wersji 6. Analiza wyników pozwala poznać wydajność małych aplikacji oraz czasy przetwarzań pakietów operacji w zależności od wybranego szkieletu programistycznego.

3.1. Cel badania

Celem badania jest zweryfikowanie hipotezy dotyczącej lepszej wydajności szkieletu programistycznego Laravel w kontekście różnego obciążenia platformy w porównaniu z Symfony. Zasymulowano rzeczywisty ruch na platformie poprzez wykonywanie pojedynczych operacji, jak i dużej liczby identycznych operacji realizowanych jednocześnie. Uzyskane wyniki pozwoliły na obliczenie średnich czasów wykonywania poszczególnych operacji na pojedyncze żądanie.

3.2. Metoda badawcza

W ramach badania wykonano operacje dodawania, usuwania, edytowania czy pobierania postu blogowego i dokonywano pomiarów czasu wykonania. Dodatkowo, zasymulowano zwiększony ruch na platformie poprzez zbiorowe operacje polegające na wysyłani dziesiątek oraz setek żądań jednocześnie do aplikacji opracowanych przy użyciu obu szkieletów programistycznych. Wszystkie operacje powtórzone zostały 10-krotnie w celu osiągnięcia jak najbardziej rzeczywistych rezultatów. Zebrane wyniki czasów posłużyły do wyliczenia średniej arytmetycznej, będącej miarą średniego czasu wykonania się poszczególnych operacji.

3.3. Środowisko testowe

W celu uzyskania jak najbardziej wiarygodnych wyników badań wydajnościowych obu technologii, aplikacje uruchomiono w jednym wspólnym środowisku testowym, korzystając przy tym z pakietu XAMPP. Obie aplikacje wykorzystywały tę samą bazę danych MySQL.

Tabela 1: Konfiguracja środowiska testowego

Urządzenie	Laptop Dell Inspiron 15-3567
Procesor	Intel Core i7-7500U 2.9 GHz
Pamięć RAM	8 GB
System operacyjny	Windows 10 Home 64-bitowy
Baza danych	MySQL 8.0
PHP	8.2
XAMPP	8.2.4
Serwer Apache	2.4

Czasy wykonywania operacji mierzone były przy użyciu narzędzia Postman. Po zainicjowaniu żądania, Postman rejestruje moment wysłania zapytania oraz moment otrzymania odpowiedzi. Mając te wartości obliczana jest różnica, która odpowiada czasowi wykonania operacji.

3.4. Opis eksperymentu

W celu wykonania analizy wydajnościowej szkieletów programistycznych Laravel 10 oraz Symfony 6 na podstawie aplikacji realizujących operacje CRUD, wykonano następujące czynności:

1. Skonfigurowanie lokalnego środowiska lokalnego

Przeprowadzono proces konfiguracji środowiska lokalnego, uwzględniając specyfikacje techniczne obu szkieletów programistycznych. Zapewniono odpowiednie ustawienia bazy danych i narzędzi niezbędnych do prawidłowego działania aplikacji.

2. Wykonanie pojedynczych żądań realizujących operacje na bazie danych

Zrealizowano powtórzone dziesięciokrotnie operacje dodawania, edytowania, usuwania i pobierania postów blogowych przy użyciu obu aplikacji testowych. Zebrano czasy wykonania każdej operacji w celu późniejszej analizy danych.

3. Zasymulowanie zbiorowych operacji dodawania danych do bazy

Przeprowadzono, powtarzane dziesięciokrotnie, symulacje wielokrotnego dodawania postów. Operacje te polegały na jednoczesnym wysyłaniu dziesiątek oraz setek żądań do aplikacji testowych, realizujących zapis danych do bazy. Czasy wykonania tych operacji zostały zgromadzone do oceny wydajności w warunkach zwiększonego ruchu na platformie.

4. Zasymulowanie operacji pobierania dużych zestawów danych

Przeprowadzono powtarzane 10-krotnie symulacje polegającą na zweryfikowaniu jak poszczególne szkielety programistyczne radzą sobie ze zwiększającą się ilością danych do pobrania. Operacja polegała na pobraniu stu, tysiąca oraz dziesięciu tysięcy postów blogowych jednocześnie.

5. Zebranie wyników

Po przeprowadzeniu symulacji zebrano wyniki czasów wykonania operacji dla każdego badanego scenariusza, dla każdego szkieletu programistycznego. Zastosowano narzędzie Postmam do pomiaru czasów wykonywania operacji. Czas ten liczony był od momentu wysłania żądania do serwera, do momentu otrzymania odpowiedzi.

6. Uśrednienie i analiza wyników oraz ich graficzna prezentacja

Przeprowadzono analizę zebranych wyników wraz z wyliczeniem średniego czasu wykonywania się poszczególnych operacji. Następnie utworzono wykresy prezentujące wizualnie różnice w czasach wykonania operacji dla obu szkieletów programistycznych.

7. Sformułowanie wniosków

Na podstawie przeprowadzonej analizy, sformułowano wnioski dotyczące wydajności testowanych szkieletów programistycznych.

3.5. Obiekt badań

Utworzone aplikacje testowe wykorzystują darmową bazę danych MySQL. Baza danych *daily_blog* zawiera tabelę *users*, która przechowuje informacje o zarejestrowanych użytkownikach w systemie oraz tabelę *posts*, zawierającą wszystkie dodane przez nich posty blogowe. Tabela *users* połączona jest z tabelą *posts* relacją 1:n.

W celu zapewnienia bezpieczeństwa przechowywanych danych zaimplementowano mechanizm szyfrowania haseł przy użyciu algorytmu Bcrypt. Kolumny *email*, *country*, *town* oraz *category* posiadają indeksy, w celu zwiększenia wydajności bazy danych. W ramach eksperymentu przeprowadzono testy jednostkowe, obejmujące kluczowe funkcje i ścieżki obu aplikacjach. Testy te miały na celu zweryfikowanie poprawności implementacji oraz identyfikację potencjalnych błędów.

4. Wyniki

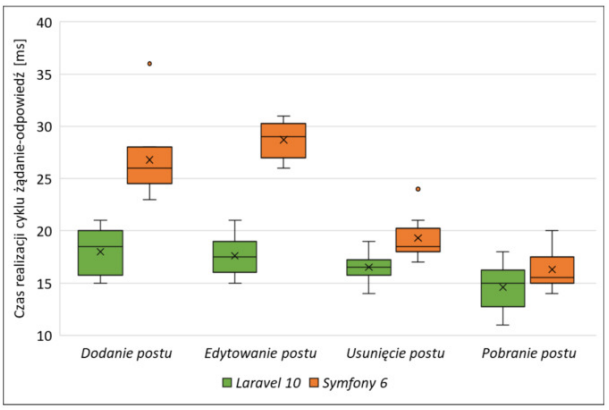
W wyniku przeprowadzonych badań wydajnościowych szkieletów programistycznych Laravel 10 i Symfony 6, uzyskano szereg wyników. Wszystkie operacje w obu aplikacjach zostały powtórzone 10-krotnie, w celu obliczenia średniego czasu wykonywania poszczególnej akcji. W przypadku pojedynczych operacji, takich jak dodawanie czy edytowanie pojedynczego postu, to Laravel 10 zaprezentował się bardziej efektywnie niż Symfony 6.

- Dodanie postu w Laravelu zajęło średnio 18 ms, podczas gdy w Symfony było to 27 ms.
- Edycja postu w Laravelu zajęła średnio 18 ms, podczas gdy w Symfony było to 29 ms.

Podczas operacji usuwania oraz pobierania, różnice były mniej zauważalne, choć nadal Laravel w wersji 10 przetwarzał te operacje szybciej.

- Usunięcie postu w Laravelu zajęło średnio 17 ms, podczas gdy w Symfony było to 19 ms.
- Wyświetlenie postu w Laravelu zajęło średnio 15 ms, podczas gdy w Symfony było to 16 ms.

Przedstawione wcześniej wyniki operacji CRUD zostały przedstawione na wykresie pudełkowym na rysunku 2. Linia ciągła znajdująca się wewnątrz każdego prostokąta przedstawia medianę wartości, natomiast krzyżyk oznacza średnią. Górna pozioma kreska ukazuje wartość maksymalną, a dolna wartość minimalną. Dla operacji dodawania oraz usuwania postu możemy dostrzec wartości odstające, które zostały ukazane w postaci kropek znajdujących się nad pudełkiem. Wartości odstające pojawiły się w przypadku szkieletu programistycznego Symfony 6.



Rysunek 2: Średni czas wykonywania pojedynczej operacji typu CRUD.

Wraz z rosnącą liczbą wykonywanych opeacji, zauważalna jest lepsza wydajność szkieletu programistycznego Symfony 6 niż Laravel 10. Podczas jednoczesnego dodawania 10 postów blogowych, różnica w średnim czasie wykonania operacji wynosi nieco mniej niż 80 milisekund (Rysunek 3). Laravel w wersji 10 wymaga zatem około 47% więcej czasu na wykonanie tej samej akcji. Wraz ze wzrostem ilości wykonywanych operacji, wydajność szkieletu programistycznego Laravel 10 maleje w porównaniu do Symfony w wersji 6. Jednoczesne dodanie 100 postów blogowych zajmuje mu średnio o 600 milisekund dłużej (Rysunek 4). W przypadku 500 żądań różnica wynosi około 3,5 sekundy (Rysunek 5), a w przypadku 1000 żądań różnica ta wynosi już niemal 7 sekund (Rysunek

6). Różnice na korzyść Symfony 6 widoczne są również podczas zbiorowych żądań pobierania, usuwania oraz edytowania. Laravel potrzebuje prawie 160 ms więcej czasu na wykonanie 10 krotnego pobierania postu, 135 ms więcej czasu na usunięcie 10 postów blogowych oraz około 65 ms więcej czasu na 10 krotne edytowanie postu blogowego (Rysunek 3). Wraz ze wzrostem ilości przetwarzanych żądań różnice w wydajności są zauważalne coraz mocniej. Na podstawie rysunku 4 można zauważyć, iż Laravel 10 potrzebował około:

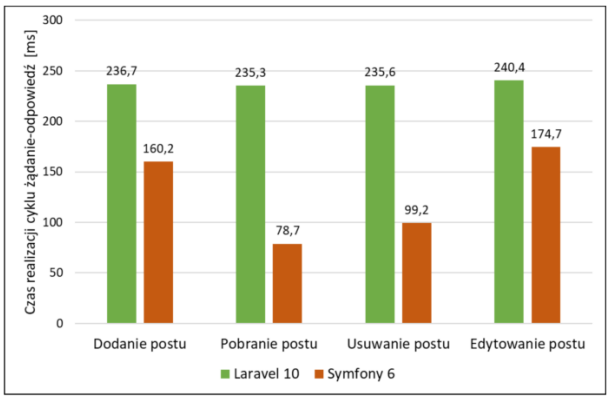
- 1600 ms więcej czasu na przetworzenie 100 żądań pobrania postu,
- 130% więcej czasu na wykonanie 100 żądań usuwania postu blogowego,
- 600 ms więcej czasu na zrealizowanie 100 żądań edytowania postu.

Na rysunku 5 widoczne są jeszcze większe różnice czasowe między operacjami typu CRUD opierające się na przetwarzaniu 500 żądań jednocześnie. Na jego podstawie można zauważyć, iż Symfony 6 potrzebowało około:

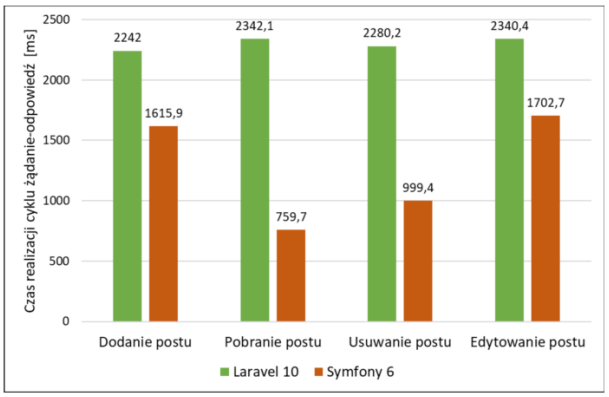
- 8 sekund mniej na wykonanie 500 żądań pobrania postu,
- 67% mniej czasu na przetworzenie 500 żądań usuwania postu blogowego,
- 3,5 sekundy mniej czasu na zrealizowanie 500 żadań edytowania postu.

Rysunek 6 przedstawia średni czas wykonania 1000 żądań operacji tyu CRUD. Na jego podstawie można zauważyć, iż Laravel 10 potrzebował około:

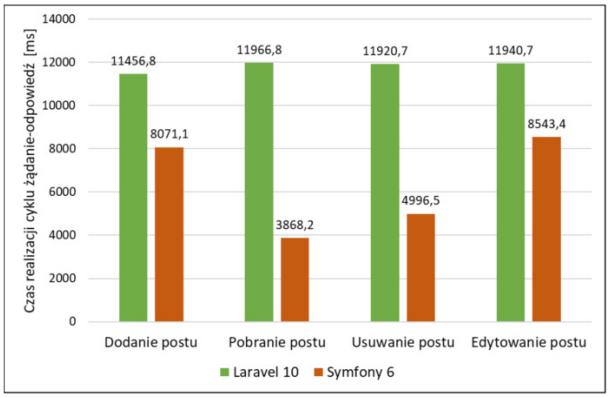
- 19 sekund więcej na przetworzenie 1000 żądań pobrania postu,
- 150% więcej czasu na wykonanie 1000 żądań usuwania postu blogowego,
- 8,5 sekundy więcej czasu na zrealizowanie 1000 żądań edytowania postu.



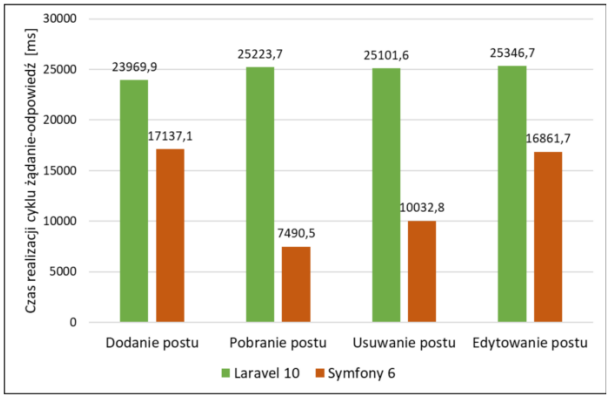
Rysunek 3: Średni czas wykonania 10 żądań operacji typu CRUD.



Rysunek 4: Średni czas wykonania 100 żądań operacji typu CRUD.

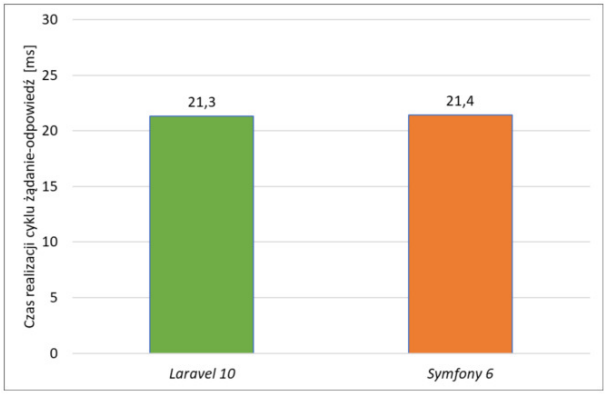


Rysunek 5: Średni czas wykonania 500 żądań operacji typu CRUD.

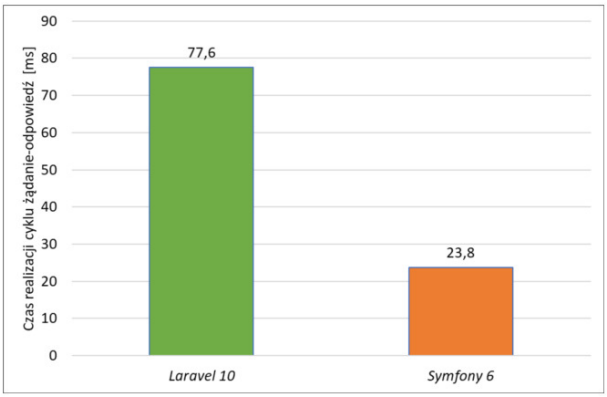


Rysunek 6: Średni czas wykonania 1000 żądań operacji typu CRUD.

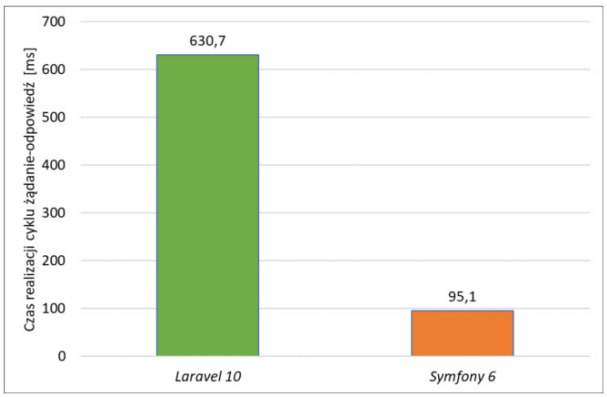
Różnice na korzyść Symfony 6 widoczne są również podczas pobierania coraz to większych zestawów danych. Wraz ze wzrostem ilości pobieranych postów blogowych wydajność Laravela 10 spada derastycznie. Pobieranie 1000 postów blogowych zajmuje mu ponad 3 razy więcej czasu (Rysunek 8) w porównaniu z drugim szkieletem programistycznym, a w przypadku pobierania 10000 postów różnica ta jest już prawie 9 krotnie większa (Rysunek 9). Jedynie podczas pobierania 100 postów blogowych (Rysunek 7), oba szkielety programistyczne poradziały sobie tak samo, ze względu na prawie identyczny czas przetwarzania operacji.



Rysunek 7: Średni czas wykonania operacji jednoczesnego pobierana 100 postów.



Rysunek 8: Średni czas wykonania operacji jednoczesnego pobierania 1000 postów.



Rysunek 9: Średni czas wykonania operacji jednoczesnego pobierania 10000 postów.

Analiza wyników pokazuje, iż podczs operacji typu CRUD Laravel 10 charakteryzuje się krótszym czasem odpowiedzi, co sugeruje lepszą wydajność tego narzędzia w sytuacjach, gdzie istotna jest szybkość obsługi pojedynczych akcji. Symfony 6 wyróżnia się szybszym czasem przetwarzania większej ilości operacji oraz pobieraniem dużych zestawów danych.

5. Wnioski

W dzisiejszych czasach postęp technologiczny z roku na rok jest coraz bardziej zauważalny. Firmy starają się coraz mocniej nadążać za postępem technologicznym, cyfryzacją czy przenoszeniem swojej sprzedaży na platformy internetowe. Ten dynamiczny rozwój powoduje stawianie coraz większych wymagań dla aplikacji internetowych, a wybór odpowiedniej technologii staje się coraz trudniejszy ze względu na dostępność coraz większej liczby narzędzi.

Głównym celem pracy było określenie, który z analizowanych szkieletów programistycznych – Laravel 10 czy Symfony 6 będzie lepszym rozwiązaniem w przypadku tworzenia małych aplikacji. Wydajność została przebadana w oparciu o wcześniej zdefiniowane kryteria badawcze. Na podstawie zebranych wyników można stwierdzić, że Symfony 6 jest lepszym rozwiązaniem dla niewielkich aplikacji typu CRUD, gdzie brana pod uwagę jest wydajność aplikacji.

Analiza wykazała, że Symfony 6 w przypadku pojedynczych operacji radzi sobie znacznie gorzej niż jego konkurent. Większość operacji zajmowała mu kilka bądź kilkadziesiąt milisekund więcej. Odwrotne zjawisko widoczne jest podczas pobierania większych zasobów danych bądź potrzeby wykonania kilkudziesięciu bądź kilkuset żądań jednocześnie. Symfony 6 staje się wtedy dużo bardziej wydajnym szkieletem programistycznym, a jego przewaga względem Laravela 10 rośnie znacząco.

Wyżej ukazane wyniki badań pokazują, iż oba frameworki są narzędziami wydajnymi ze względu na szybkość przetwarzania żądań. Decyzja między Laravel a Symfony powinna być ściśle uzależniona od wymagań projektowych oraz charakterystyki aplikacji.

Literatura

[1] M. Laaziri, K. Benmoussa, S. Khoulji, K. M. Larbi, A. El Yamami, A comparative study of Laravel and Symfony PHP frameworks, International Journal of Electrical and Computer Engineering 9(1) (2019) 704–712, http://doi.org/10.11591/ijece.v9i1.pp704-712.

[2] P. Garbarz, M. Plechawska-Wójcik, Comparative analysis of PHP frameworks on the example of Laravel and Symfony, Journal of Computer Sciences Institute 22 (2022) 18–25, https://doi.org/10.35784/jcsi.2781.

[3] Biztech, https://www.biztechcs.com/blog/laravel-vs-symfony-which-is-the-better-framework-in-2024, [09.04.2024].

[4] N. Yadav, D. S. Rajpoot, S. K. Dhakad, LARAVEL: a PHP framework for e-commerce website, In 2019 Fifth International Conference on Image Information Processing (ICIIP) (2019) 503–508, http://doi.org/10.1109/iciip47207.2019.8985771.

[5] 7 Most Popular PHP Frameworks To Use in 2024, https://www.excellentwebworld.com/best-php-frameworks/, [28.10.2023].

[6] M. Laaziri, K. Benmoussa, S. Khoulji, M. L. Kerkeb, A Comparative study of PHP frameworks performance, Procedia Manufacturing 32 (2019) 864–871, http://doi.org/10.1016/j.promfg.2019.02.295.

[7] X. Li, S. Karnan, J. A. Chishti, An empirical study of three PHP frameworks, In 4th International Conference on Systems and Informatics (ICSAI) (2017) 1636–1640, http://doi.org/10.1109/icsai.2017.8248546.