JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 28 (2023) 256–263
Received: 8 June 2023
Accepted: 29 June 2023

Comparative analysis of Angular and React frameworks

Analiza porównawcza szkieletów programistycznych Angular i React

Sylwester Skrzypiec*, Małgorzata Plechawska-Wójcik

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

Abstract
This thesis aimed to examine the Angular framework and React library in terms of performance on three popular web browsers. To achieve this, two similar test applications were implemented to create user interfaces, along with a server application for database communication and the aforementioned programming frameworks. Subsequently, tests were conducted to assess the time required for downloading and displaying images, records, and their deletion. In addition to performance results, a literature review was conducted to provide a better understanding of the current state of knowledge in the field, and individual features distinguishing the respective framework or library were described. Furthermore, the obtained data was analyzed for statistical differences using statistical tests.

Keywords: Angular; React; performance; SPA; benchmarking

Streszczenie
Niniejsza praca dyplomowa miała na celu sprawdzenie szkieletu Angular oraz biblioteki React pod kątem wydajności na trzech popularnych przeglądarkach internetowych. W związku z tym zaimplementowano dwie podobne aplikacje testowe tworzące interfejsy użytkownika oraz jedną aplikację serwerową do komunikacji z bazą danych i wymienionymi szkieletami programistycznymi. Następnie zostały wykonane testy sprawdzające czas pobierania i wyświetlania danych oraz ich usuwania. Poza wynikami wydajności przeprowadzono przegląd literatury w celu lepszego zobrazowania obecnego stanu wiedzy w danym temacie oraz opisano poszczególne elementy wyróżniające dany szkielet lub bibliotekę. Sprawdzono również otrzymane dane wynikowe pod względem różnic statystycznych korzystając z testów statystycznych.

Słowa kluczowe: Angular; React; wydajność; SPA; analiza porównawcza

*Corresponding author
Email address: sylwester.skrzypiec@pollub.edu.pl (S. Skrzypiec)

1. Wstęp

W obecnych czasach tworzenie nowoczesnych systemów informatycznych ciągle stawia przed programistami nowe wyzwania, żeby sprostać rosnącym oczekiwaniom użytkowników. Żeby zaoszczędzić ich czas oraz ułatwić sobie pracę, w obecnych rozwiązaniach stosuje się architekturę SPA (*ang. Single Page Application*), która wypierając poprzednią tzn. MPA (*ang. Multi Page Application*) pozwoliła na wdrożenie nowszych, znacznie skuteczniejszych rozwiązań [1, 2, 3]. W odróżnieniu od poprzedniej, SPA pobiera wszelkie zasoby tylko przy pierwszym wejściu na daną stronę, a następnie reagując na działania użytkownika, podmienia elementy DOM (*ang. Document Object Model*) korzystając z javascript'u. [4, 5].

Język javascript pomimo swojej wszechstronności oraz popularności [6] razem z CSS (*ang. Cascading Style Sheets*) oraz HTML (ang. *HyperText Markup Language*) coraz rzadziej występuje w sposób natywny w implementacji nowoczesnych aplikacji internetowych i coraz częściej jest zastępowany przez swój nadzbiór, czyli typescript [7]. Plik typescript'owy korzysta z lekko zmienionej składni oraz są na niego nałożone dodatkowe reguły syntaktyczne, a w momencie kompilacji jest tłumaczony na język javascript, dzięki temu przeglądarki nie mają problemu z jego zrozumieniem [8]. Rozszerzenie takie pozwoliło wyeliminować niedoskonałości w postaci braku kontroli nad typami zmiennych oraz dołożyło nowe możliwości w asynchronicznej komunikacji z serwerem. Programiści oprócz wspomnianej zmiany, zaczęli również korzystać z tak zwanych szkieletów programistycznych. Obecnie topowe miejsca zajmują dwa z nich: Angular oraz React [9].

Angular jest szkieletem rozwijanym przez firmę Google oferując deweloperowi wszelkie narzędzia do tworzenia aplikacji internetowych [10]. Używa on swoich rozwiązań w postaci wstrzykiwania zależności, bibloteki „Material" do oprawy graficznej interfejsu oraz zapewnia czysty oraz testowalny kod. Poza tym umożliwia sprawne zarządzanie dostępem do poszczególnych stron, dzięki użyciu komponentów jest możliwe używanie tych samych szablonów HTML w wielu miejscach, przez co same aplikacje są bardziej otwarte na rozbudowę oraz pozwalają na bezkonfliktowe zmiany.

React jest rozwijany przez firmę Facebook oraz w porównaniu do Angular'a jest bibloteką javascript'u a nie szkieletem programistycznym [11]. Jego składnia opiera się na JSX *(ang. Javascript Extensible Markup Language*), które podobnie jak typescript jest rozszerzeniem javascript'u [12]. Takie rozwiązanie oferuje przyspieszoną kompilację kodu oraz czyni kod bardziej czytelnym i łatwiejszym do utrzymania [13].

W związku z szeroką ofertą dostępnych narzędzi w omówionych wyżej rozwiązaniach, zdecydowano się porównać oba szkielety pod kątem wydajności w pobie-

raniu oraz wyświetlaniu danych na przeglądarce Google Chrome.

2. Przegląd literatury

W celu lepszego zrozumienia badanego problemu, przed przystąpieniem do badań został wykonany przegląd literatury nawiązującej do opracowanego tematu.

Pierwszym analizowanym tekstem została praca dyplomowa pt. „Evaluation of Front-end JavaScript Frameworks for Master Data Management Application Development" [14]. Dotyczy ona porównania szkieletów programistycznych do tworzenia interfejsu użytkownika w aspekcie zarządzania danymi podstawowymi (*ang. Master Data Management, MDM*). Autor analizuje wszelkie możliwości szkieletów, architektury, dane statystyczne oraz bada je pod kątem szybkości wykonywania operacji typu CRUD (*ang. create, read, update, delete, tłum. utwórz, odczytaj, aktualizuj, usuń*). Analizując wyniki badań oraz wnioski pracy odkryto, że najlepszym wyborem do stworzenia aplikacji typu MDM jest Angular ze względu na przewagę nad React i Vue w kwestiach popularności, szybkości kompilacji czy możliwości dostosowywania języka interfejsu graficznego do użytkownika.

Inną przydatną pozycją literaturową okazała się praca dyplomowa pt. „Development of an evaluation model for client-side JavaScript Frameworks" [15]. Autor skupia się na przedstawieniu kryteriów oraz oceny poszczególnych szkieletów, żeby wspomóc programistów w ich wyborze. Do porównania przyjęto trzy najpopularniejsze z nich tzn. Angular, React oraz Vue. W kwestii kryteriów oceny przyjęto wszelkie istotne kwestie związane z wytwarzaniem oprogramowania, czyli łatwością implementacji wymagań funkcjonalnych na przykład możliwość wykorzystywania komponentów do budowy interfejsu użytkownika. Do innych ważnych czynników zaliczono wymagania niefunkcjonalne jak na przykład kwestia bezpieczeństwa, stabilności lub wydajności. Poza przeprowadzonymi badaniami, autor opisał również archtiektury szkieletów oraz opisał możliwości poszczególnych szkieletów. Analizując przedstawione wnioski ustalono, że pierwsze miejsce zajeli React oraz Vue z wynikiem 23/28 punktów. Angular otrzymał 22/28, więc można stwierdzić, że wszystkie trzy możliwości z pewnością nadają się do implementacji interfejsu użytkownika i pomimo występujących różnic w dostępnych narzędziach, poradzą sobie z wszelkimi wymaganiami.

Autorzy pracy dyplomowej pt. „JavaScript Frameworks A qualitative evaluation and comparison of the dominant factors in Angular and React" [16] skoncentrowali się na udzieleniu pomocy początkującym programistom w wyborze szkieletu programistycznego między Angular'em a React'em, odpowiedniego do ich potrzeb. Na początku, zostały opisane poszczególne narzędzia dostępne w szkieletach oraz sam bazowy język Javascript. Następnie poprzez zastosowanie cykli kolba, jako modelu uczenia się przez doświadczenie, poddano ocenie oba szkielety. Uczenie podzielono na cztery okresy sumujące się do czasu 14 dni, gdzie w każdym okresie opisano trudności występujące podczas nauki oraz ogólne subiektywne wnioski towarzyszące implementacji. Analizując punkty przyznane w kategoriach uczenia, najlepszy okazał się Angular.

3. Cel i zakres badań

Celem prowadzonych badań było porównanie wydajnościowe szkieletów programistycznych Angular oraz React. Zakres badań obejmował analizę czasową przetwarzania danych pobranych z serwera w celu potwierdzenia lub negacji tezy oraz odpowiedzi na pytania badawcze. Opracowano następującą tezę:

Szkielet programistyczny Angular jest szybszy w przetwarzaniu danych względem bibloteki React.
Pytania badawcze:

Q1: Czy szkielet Angular szybciej pobierze i wyświetli odpowiednio 1 000, 5 000, 10 000 rekordów niż React?

Q2: Czy React szybciej usunie odpowiednio 100, 500, 1 000 rekordów niż Angular?

Q3: Czy Angular szybciej pobierze i wyświetli 50, 100, 200 zdjęć w formacie .JPG?

4. Plan badań

Przed przystąpieniem do realizacji badań zostały zaimplementowane dwie aplikacje internetowe wykonane w szkieletach Angular i React oraz jedna aplikacja serwerowa obsługująca REST API (*ang. Representational state transfer application programming interface*) do obsługi żądań związanych z pobraniem i wysyłaniem danych.

Testy wydajnościowe opisane poniżej w celu lepszego sprawdzenia wydajności szkieletów, zostały wykonane na przeglądarkach: Google Chrome, Mozilla Firefox oraz Microsoft Edge.
Eksperyment zakładał następujące etapy badań:

1. Wykonanie badań:
   a. wykonanie dwudziestu pomiarów pobrania 1 000, 5 000, 10 000 rekordów z użyciem Angular'a na każdej z trzech przeglądarek,
   b. wykonanie dwudziestu pobrań 1 000, 5 000, 10 000 rekordów z użyciem React'a na każdej z trzech przeglądarek,
   c. wykonanie dwudziestu pomiarów usunięcia 100, 500, 1 000 rekordów przy użyciu Angular'a na każdej z trzech przeglądarek,
   d. wykonanie dwudziestu pomiarów usunięcia 100, 500, 1 000 rekordów z użyciem React'a na każdej z trzech przeglądarek,
   e. wykonanie dwudziestu pobrań i wyświetleń 50, 100, 200 zdjęć z użyciem Angular'a na każdej z trzech przeglądarek,
   f. wykonanie dwudziestu pobrań i wyświetleń 50, 100, 200 zdjęć z użyciem React'a na każdej z trzech przeglądarek.
2. Analiza popularności oraz dostępu do materiałów obu szkieletów:
   a. sprawdzenie liczby dostępnych materiałów bazując na słowach kluczowych związanych z danym szkieletem,

   b. analiza rankingu popularności według społeczności oraz dostępnych ankiet.
3. Analiza wyników badań
   a. przedstawienie danych w formie wykresów, tabel,
   b. wykonanie testów statystycznych i analiza ich wyników.
4. Wnioski

Specyfikacja urządzenia wykorzystanego do przeprowadzenia testów wydajnościowych została podana w Tabeli numer 1.

Tabela 1: Specyfikacja urządzenia

Komponent	Specyfikacja
System operacyjny	Windows 10 Home Wersja 22H2 x64
Procesor	Intel Core i7-7700HQ CPU 2.80 GHz
Pamięć RAM	DDR4 2400MHz 16GB
Karta graficzna	GeForce MX150, GDDR5 2 GB
Dysk	HDD SATA 5400 obr. 1TB

5. Opis wykorzystanych szkieletów

W tworzeniu aplikacji internetowych często używa się szkieletów programistycznych (*ang. frameworks*). Dzięki nim praca jest szybsza, ponieważ udostępniają gotowe narzędzia i komponenty. Szkielety pozwalają na stylizację elementów, walidację formularzy oraz manipulację elementami DOM. Przy tworzeniu aplikacji bez szkieletów programiści muszą samodzielnie zadbać o skalowalność i bezpieczeństwo systemu. Szkielety programistyczne pomagają w ochronie przed atakami takimi jak XSS, CSRF i SQL Injection, co ułatwia wprowadzanie zabezpieczeń do aplikacji [17, 18, 19].

Ogólne różnice pomiędzy badanymi szkieletami zostały przedstawione w Tabeli numer 2.

Tabela 2: Wykaz różnic pomiędzy badanymi szkieletami

Różnica	Angular	React
Firma utrzymująca	Google	Facebook
Język oraz rozszerzenia	Typescript	Javascript/Typescript/JSX
Wiązanie danych	Jedno lub dwukierunkowe	Jednokierunkowe
Typ szkieletu	Szkielet programistyczny	Biblioteka
Łatwość uczenia	Trudny	Średni
Elastyczność	Niska	Wysoka
Dodatkowe narzędzia	Większość wbudowanych	Potrzeba dodatkowej instalacji
Styl architektury	MVC	Otwarta
Społeczność	Duża	Ogromna

5.1. Szkielet Angular

Historia Angular'a sięga roku 2009, kiedy to dwóch programistów pracujących w Google stworzyło projekt o nazwie AngularJS [20]. Po wewnętrznym wdrożeniu projektu w Google, w 2012 roku AngularJS został udostępniony, jako szkielet open-source, zyskując duże poparcie społeczności programistów. Jednak wraz z pojawieniem się bardziej złożonych interfejsów użytkownika, AngularJS zaczął mieć problemy z wydajnością. W odpowiedzi na te wyzwania, Google postanowiło opracować nową wersję szkieletu o nazwie "Angular", napisaną w języku TypeScript.

TypeScript, będący nadzbiorem JavaScriptu, wprowadza statyczne typowanie zmiennych oraz obiektowość, co eliminuje przypadkowe błędy i ułatwia zarządzanie kodem aplikacji [7]. Architektura Angular'a opiera się na wzorcu projektowym Model-Widok-Kontroler (*MVC, ang. Model-View-Controller*), co umożliwia lepszą strukturę i skalowalność kodu [20]. Warstwa modelu jest zarządzana za pomocą serwisów, które komunikują się z API (*ang. Application Programming Interface, tłum. Interfejs Programowania Aplikacji*), w celu pobierania i wysyłania danych. Warstwa widoku składa się z szablonów i dyrektyw, które umożliwiają tworzenie i modyfikację elementów DOM. Kontroler pełni rolę interfejsu, łącząc model i widok, obsługując przetwarzanie danych i walidację [21].

Dzięki podziałowi na modele, widoki i kontrolery, Angular zapewnia lepsze utrzymanie kodu, łatwiejsze testowanie i skalowalność. Komponenty są hierarchicznie osadzane w głównym pliku "app.component.ts", a moduły z własnymi ścieżkami trasowania umożliwiają wyświetlanie odpowiednich komponentów w zależności od nawigacji użytkownika. Każdy komponent składa się z plików: komponentu, styli, szablonu i testów jednostkowych. Angular oferuje również testy jednostkowe oraz testy E2E, które pozwalają sprawdzić działanie aplikacji w różnych scenariuszach.

Podsumowując, Angular jest potężnym narzędziem do tworzenia zaawansowanych interfejsów użytkownika, oferującym wiele gotowych narzędzi, które przyspieszają rozwój projektów. Znajduje często zastosowanie w projektach komercyjnych i jest ciągle rozwijany przez swojego producenta.

5.2. Biblioteka React

React to biblioteka do tworzenia interfejsów użytkownika (UI). Została stworzona w 2011 roku przez inżynierów z firmy Facebook, aby rozwiązać problemy z implementacją skomplikowanych UI [12]. React umożliwia tworzenie aplikacji internetowych oraz aplikacji mobilnych za pomocą narzędzia React Native. Jest to biblioteka open-source rozwijana przez Facebooka.

Głównym konceptem React jest tworzenie wirtualnego drzewa DOM, które reprezentuje rzeczywiste drzewo DOM w przeglądarce [12, 22]. Biblioteka porównuje oba drzewa i aktualizuje tylko te elementy, które uległy zmianie, co sprawia, że jest bardzo wydajny. Można go używać z różnymi architekturami, takimi jak MVC lub MVVM, gdzie MVVM (*ang. Model-*

View-ViewModel, tłum. Model-Widok-WidokModelu) dodaje warstwę widoku modelu, izolując logikę biznesową od warstwy prezentacji.

React wykorzystuje JavaScript, jako główny język programowania, ale od niedawna można również używać TypeScript. Biblioteka oferuje różne "uchwyty stanu" (*ang. state hooks*), takie jak useState, useContext i useReducer, które umożliwiają zarządzanie stanem komponentów [23]. Stan jest magazynem danych komponentów i może się zmieniać w zależności od kontekstu, co pozwala na dynamiczne renderowanie komponentów.

Ważnym elementem React jest JSX, które jest rozszerzeniem składni JavaScript. JSX pozwala na definiowanie elementów UI w sposób podobny do HTML i jest łatwiejszy do czytania i modyfikacji niż czysty JavaScript. JSX zapewnia również bezpieczeństwo, ponieważ przed wyświetleniem elementu React stosuje "znaki ucieczki", aby uniknąć wstrzykiwania niepożądanego kodu (atak XSS [17]).

W testowaniu, React korzysta z szkieletu testowego o nazwie "Jest", który umożliwia testowanie komponentów. Można również użyć biblioteki React Testing Library, która skupia się na testowaniu interakcji użytkownika z aplikacją poprzez testowanie drzewa DOM [23].

Podsumowując, React to również świetne narzędzie do tworzenia zaawansowanych interfejsów użytkownika. Posiada wiele funkcji i narzędzi, które ułatwiają tworzenie skalowalnych i bezpiecznych aplikacji internetowych. Jest ciągle rozwijany przez społeczność programistyczną i może być wykorzystywany w projektach komercyjnych.

5.3. Inne wykorzystane narzędzia

W ramach przeprowadzonych badań wykorzystano również inne narzędzia istotne dla działania aplikacji internetowych. Warto wspomnieć o bazie danych, zapleczu technicznym oraz architekturze REST API, które są nieodzownymi elementami aplikacji internetowych.

Jednym z najczęściej wykorzystywanych narzędzi jest szkielet Spring Boot. Został on stworzony w 2014 roku, jako rozszerzenie szkieletu Java Spring, mające na celu ułatwienie konfiguracji i zapewnienie większej modularności [24]. Spring Boot umożliwia tworzenie zaplecza technicznego dla aplikacji internetowych oraz mobilnych. Posiada wsparcie dla tworzenia mikro usług oraz implementacji CMS'ów, co sprawia, że jest popularny wśród programistów.

Do przechowywania danych wykorzystano bazę danych MySQL. Jest to relacyjna baza danych, która oferuje wiele funkcjonalności, takich jak transakcje, indeksy czy widoki [25]. Poza bazodanowymi funkcjonalnościami zapewnia również narzędzie do graficznej budowy schematu relacyjnej bazy danych oraz możliwość wygenerowania kodu SQL ze schematu lub odwrotnie. MySQL jest jednym z najpopularniejszych rozwiązań i charakteryzuje się prostą konfiguracją, co ułatwia jego wykorzystanie w małych projektach.

W kontekście komunikacji między elementami systemu informatycznego, zastosowano architekturę REST API [26]. REST API wykorzystuje protokół HTTP do komunikacji między klientem a serwerem. Dane są przekazywane w postaci zasobów, najczęściej w formacie JSON (*ang. JavaScript Object Notation, tłum. Notacja Obiektu JavaScript*). Komunikacja odbywa się za pomocą standardowych metod HTTP, takich jak GET, POST, PUT i DELETE [26]. REST API zapewnia jednolity interfejs oraz zasady, które umożliwiają skalowalność i łatwą zmienność systemu.

Wykorzystanie tych narzędzi przynosi wiele korzyści w procesie tworzenia aplikacji internetowych. Spring Boot zapewnia elastyczność i skalowalność, baza danych MySQL umożliwia przechowywanie i manipulację danymi, a architektura REST API zapewnia spójną komunikację między elementami systemu.

6. Wyniki badań

Podczas wykonywania badań nałożono szczególny nacisk, aby urządzenie, na którym zostały one wykonenie było obciążone innymi usługami oprócz tych wymaganych do przeprowadzenia testów. Ze względów ograniczających dostępne zasoby, każde wywołanie zapytania do API zostało opóźnione o 5 ms, sumaryczny czas opóźniający został następnie odjęty od końcowego. Dla obu szkieletów przeprowadzono testy dwudziestokrotnie dla wszystkich przypadków na trzech popularnych przeglądarkach. Następnie obliczono podstawowe dane statystyczne w formie średniej oraz odchylenia standardowego. Poza wymienionymi danymi statystycznymi wykorzystano również testy statystyczne szerzej omówione w dalszych podrozdziałach.

6.1. Wyniki testów wydajnościowych

W Tabeli numer 3 zostały zamieszczone średnie wyniki wraz z uwzględnieniem odchylenia standardowego uzyskane dla poszczególnych badanych operacji, uzyskanych na przeglądarce Google Chrome.

Tabela 3: Wyniki testów na przeglądarce Google Chome

Próba	Angular [ms]	React [ms]
Pobranie i wyświetlenie 1 000 rekordów	10 258 ± 973	18 482 ± 1 630
Pobranie i wyświetlenie 5 000 rekordów	142 536 ± 13 163	491 189 ± 20 470
Pobranie i wyświetlenie 10 000 rekordów	603 473 ± 30 770	1 928 329 ± 79 178
Pobranie i wyświetlenie 50 zdjęć	1 832 ± 270	3 418 ± 322
Pobranie i wyświetlenie 100 zdjęć	3 829 ± 372	8 862 ± 327
Pobranie i wyświetlenie 200 zdjęć	5 725 ± 693	26 370 ± 2 494
Usunięcie 100 rekordów	598 ± 89	438 ± 36

Usunięcie 500 rekordów	6 203 ± 320	8 758 ± 247
Usunięcie 1 000 rekordów	18 431 ± 994	32 842 ± 805

W Tabeli numer 4 przedstawiono analogicznie wyniki testów wydajnościowych, wykonanych na przeglądarce Mozilla Firefox.

Tabela 4: Wyniki testów na przeglądarce Mozilla Firefox

Próba	Angular [ms]	React [ms]
Pobranie i wyświetlenie 1 000 rekordów	13 881 ± 389	25 409 ± 679
Pobranie i wyświetlenie 5 000 rekordów	195 851 ± 15 109	524 966 ± 23 173
Pobranie i wyświetlenie 10 000 rekordów	795 304 ± 81 189	2 206 582 ± 70 321
Pobranie i wyświetlenie 50 zdjęć	2 262 ± 243	3 602 ± 165
Pobranie i wyświetlenie 100 zdjęć	4 199 ± 197	8 590 ± 248
Pobranie i wyświetlenie 200 zdjęć	6 919 ± 928	22 978 ± 1 228
Usunięcie 100 rekordów	1 084 ± 16	1 121 ± 80
Usunięcie 500 rekordów	5 740 ± 197	6 722 ± 113
Usunięcie 1 000 rekordów	15 866 ± 659	25 486 ± 819

W przypadku testów wykonanych na przeglądarce Microsoft Edge, wyniki zostały przedstawione w Tabeli numer 5.

Tabela 5: Wyniki testów na przeglądarce Microsoft Edge

Próba	Angular [ms]	React [ms]
Pobranie i wyświetlenie 1 000 rekordów	10 704 ± 1 387	18 472 ± 1 710
Pobranie i wyświetlenie 5 000 rekordów	128 164 ± 1 805	479 991 ± 7 871
Pobranie i wyświetlenie 10 000 rekordów	536 469 ± 6 279	1 984 726 ± 32 450
Pobranie i wyświetlenie 50 zdjęć	2 961 ± 574	4 258 ± 287
Pobranie i wyświetlenie 100 zdjęć	4 310 ± 398	10 304 ± 426
Pobranie i wyświetlenie 200 zdjęć	7 432 ± 755	28 742 ± 945
Usunięcie 100 rekordów	738 ± 111	492 ± 61
Usunięcie 500 rekordów	6 803 ± 611	8 637 ± 333
Usunięcie 1 000 rekordów	23 892 ± 778	31 682 ± 663

Podczas porównywania wyników, warto zauważyć wysoce nieliniowy charakter uzyskanych danych, który jest związany ze zwiększeniem obciążenia spowodowanego liczbą przetwarzanych danych.

Podsumowując wyniki uzyskane na poszczególnych przeglądarkach dla danych typów badanych operacji zdecydowano się na dodatkowe przedstawienie w formie tabeli szkieletów, które szybciej poradziły sobie z zadaniem. Obserwując wyniki pokazane w Tabeli 6 można zauważyć znaczną przewagę szkieletu Angular (kolor czerwony) nad biblioteką React (kolor zielony) w zdecydowanej większości badanych operacji. Bazując na średnich wynikach pojedynczych operacji zestawionych w Tabelach 3 – 5, podano szkielet, który szybciej poradził sobie z wykonaniem zadanego zadania. W 25/27 operacji Angular okazał się lepszy pod względem czasowym.

Tabela 6: Wykaz sumaryczny testów wydajnościowych

	Chrome	Firefox	Edge
Pobranie i wyświetlenie 1 000 rekordów	Angular	Angular	Angular
Pobranie i wyświetlenie 5 000 rekordów	Angular	Angular	Angular
Pobranie i wyświetlenie 10 000 rekordów	Angular	Angular	Angular
Pobranie i wyświetlenie 50 zdjęć	Angular	Angular	Angular
Pobranie i wyświetlenie 100 zdjęć	Angular	Angular	Angular
Pobranie i wyświetlenie 200 zdjęć	Angular	Angular	Angular
Usunięcie 100 rekordów	React	Angular	React
Usunięcie 500 rekordów	Angular	Angular	Angular
Usunięcie 1 000 rekordów	Angular	Angular	Angular
Ocena punktowa	8:1	9:0	8:1
Sumarycznie	25:2		

6.2. Wyniki testów statystycznych

Jak wspomniano poza wykorzystaniem podstawowych danych statystycznych zdecydowano się na użycie testów statystycznych. Wykorzystane testy pozwolą potwierdzić lub odrzucić hipotezę domyślną - h0, mówiącą czy istnieje statystycznie istotna różnica pomiędzy badanymi grupami próbek.

W związku z decyzją o wykorzystaniu testów t-student'a, Mann'a-Whitney'a w przypadku porównania identycznych operacji wykonanych dla tej samej przeglądarki wykorzystano testy Shapiro-Wilk'a oraz Levene'a w celu sprawdzenia zgodności danych z rozkładem

normalnym oraz homogeniczności wariancji (warunek skorzystania z testu t-student'a) [27]. Drugim badaniem z wykorzystaniem testów Anova oraz Kruskal'a-Wallis'a było porównanie danych uzyskanych dla jednego szkieletu na wszystkich przeglądarkach [28]. Również w tym przypadku posiłkowano się danymi z testów Shapiro-Wilk'a oraz Levene'a w celu sprawdzenia warunków koniecznych do użycia testu Anova [29]. Ustalono stały poziom istotności dla wszystkich testów na poziomie $\alpha = 0.05$.

W Tabeli numer 7 zostały przedstawione wyniki testów t-student'a (kolor zielony) oraz Mann'a-Whitney'a. Testy wykonywane były w zależności od wartości p-value uzyskanych z testów Shapiro-Wilk'a oraz Levene'a wykonanych na poszczególnych grupach danych. Warto przypomnieć, że w przypadku tego testu porównywane były wyniki uzyskane dla identycznych operacji z wykorzystaniem osobnych szkieletów.

Tabela 7: Wyniki testów t-student'a oraz Mann'a-Whitney'a

Operacja	Chrome [p-value]	Firefox [p-value]	Edge [p-value]
Wyświetlenie 1000 rekordów	6,796E-08	6,796E-08	6,786E-08
Wyświetlenie 5000 rekordów	6,796E-08	6,796E-08	6,796E-08
Wyświetlenie 10 000 rekordów	6,796E-08	6,796E-08	6,796E-08
Wyświetlenie 50 zdjęć	7,451E-19	6,786E-08	9,160E-08
Wyświetlenie 100 zdjęć	6,786E-08	2,553E-39	1,721E-34
Wyświetlenie 200 zdjęć	6,796E-08	6,796E-08	6,796E-08
Usunięcie 100 rekordów	1,425E-07	1,464E-03	2,556E-07
Usunięcie 500 rekordów	6,796E-08	6,796E-08	4,533E-07
Usunięcie 1 000 rekordów	5,526E-36	1,307E-32	6,796E-08

Poza porównaniem między-szkieletowym, zdecydowano również o sprawdzenie szkieletów pod kątem zmiany przeglądarki. W Tabeli numer 7 przedstawiono wyniki testów Anova (kolor zielony) oraz Kruskal'a-Wallis'a. Testy wykonywano również warunkowo jak w przypadku wyników z Tabeli numer 8.

Tabela 8: Wyniki testów Anova oraz Kruskal'a-Wallis'a

Operacja	Angular [p-value]	React [p-value]
Wyświetlenie 1000 rekordów	1,278E-08	2,829E-09
Wyświetlenie 5000 rekordów	9,265E-12	1,908E-07
Wyświetlenie 10 000 rekordów	1,412E-11	1,002E-09
Wyświetlenie 50 zdjęć	2,296E-08	3,738E-09
Wyświetlenie 100 zdjęć	2,586E-04	3,527E-10
Wyświetlenie 200 zdjęć	5,216E-08	8,497E-11
Usunięcie 100 rekordów	1,019E-10	3,840E-10
Usunięcie 500 rekordów	7,756E-09	1,401E-09
Usunięcie 1 000 rekordów	5,266E-12	5,115E-37

7. Porównanie statystycznych danych popularnościowych

Poza testami wydajnościowymi i statystycznymi, przeanalizowano rankingi, wsparcie społeczności i dostępną dokumentację. Sprawdzenie wsparcia społeczności na popularnym forum StackOverflow jest ważne, ponieważ programiści często szukają tam rozwiązań dla swoich problemów. Sprawdzając liczbę tematów pod słowem kluczowym „reactjs", znaleziono 456 082 tematów, z drugiej strony pod etykietą „angular" 295 979 (z dnia 29.05.2023).

W przypadku chęci sprawdzenia liczby wyników wyszukiwania dotyczących dokumentacji, artykułów i stron poświęconych danemu szkieletowi programistycznemu, można wykorzystać licznik wbudowany do przeglądarki. Przykładowo, fraza "Angular documentation" generuje 151 000 000 wyników, podczas gdy fraza "React documentation" generuje 461 000 000 wyników (z dnia 29.05.2023).

Według ankiety StateOfJs przeprowadzonej w 2022 roku i obejmującej 39 472 odpowiedzi, React cieszy się największą popularnością wśród programistów szkieletów używanych do tworzenia interfejsu użytkownika w kategorii chęci dalszego użycia [30]. Od 2016 roku widać wyraźną przewagę Reacta, który od wyniku 52,7% wzrósł do obecnego poziomu 81,8%. Angular zajmuje drugie miejsce, choć jego popularność powoli spada na rzecz szkieletu Vue.js.

8. Wnioski

Analizując powyższą pracę można zauważyć, że oba szkielety programistyczne są niezwykle użytecznymi narzędziami przy tworzeniu zaawansowanego interfejsu użytkownika. Tak jak było to wspomniane w poprzednich rozdziałach, obecne aplikacje internetowe wymagają obsługi skomplikowanej logiki biznesowej oraz asynchronicznego przetwarzania danych. Poza tym, programiści zaczęli zwracać szczególną uwagę na czystość kodu oraz jak najwyższy poziom skalowalności pomagający sprostać ciągłemu rozwojowi aplikacji.

Pomimo licznych różnic, można zauważyć, że zarówno React jak i Angular są wszechstronnymi narzędziami i mogą zostać wykorzystane w celu rozwiązania praktycznie każdego problemu w kwestii wytwarzania interfejsu użytkownika. Największą ich różnicą jest sposób implementacji poszczególnych funkcjonalności. React korzysta ze swojego JSX wstrzykiwanego wraz z zawartym w nim HTML do wirtualnego drzewa DOM, natomiast Angular jest w stanie dodawać i modyfikować kontent przy pomocy różnego rodzaju dyrektyw.

Analizując wyniki poszczególnych testów wydajnościowych możemy zauważyć, że poza operacją usuwania 100 rekordów na przeglądarkach Chrome oraz Edge, w reszcie przypadków zwycięzcą okazał się Angular. W przypadku zwiększania liczby przetwarzanych danych czas rósł nieliniowo dla obu szkieletów. Poza porównywaniem między sobą Angulara i React, warto spojrzeć również w kontekście zachowania się danego narzędzia na poszczególnej przeglądarce. Angular najlepiej poradził sobie z przetwarzaniem obrazów na przeglądarce Chrome, natomiast React na Firefox. W przypadku pobierania rekordów, oba szkielety uzyskały najlepsze wyniki na przeglądarce Edge, natomiast dla usuwania rekordów na przeglądarce Firefox.

Testy statystyczne wykazały, że pomimo częstej zgodności z rozkładem normalnym lub spełnienia warunków homogeniczności wariancji, wyniki porównywane pomiędzy szkieletami lub wewnętrznie pomiędzy przeglądarkami wykazały w każdym przypadku statystycznie istotne różnice.

Obserwując dostępne statystyki trendów chęci użycia szkieletu, widać, że React silnie dominuje opinię programistów począwszy od 2016 r. Zaraz za nim pojawia się Angular, którym deweloperzy interesują się coraz mniej. Można z tego wywnioskować, że istnieje możliwość dalszego wzrostu użycia na rynku biblioteki React oraz konsekwentny spadek popularności szkieletu Angular.

Powracając do postawionej na początku tezy - „Szkielet programistyczny Angular jest szybszy w przetwarzaniu danych względem biblioteki React", można stanowczo powiedzieć, że została ona potwierdzona ze względu na wyniki testów wydajności. Poza potwierdzeniem głównej tezy można również odpowiedzieć na zadane pytania badawcze:

- Q1: Czy szkielet Angular szybciej pobierze i wyświetli odpowiednio 1 000, 5 000, 10 000 rekordów niż biblioteka React?
- Q2: Czy biblioteka React szybciej usunie odpowiednio 100, 500, 1 000 rekordów niż szkielet Angular?
- Q3: Czy szkielet Angular szybciej pobierze i wyświetli 50, 100, 200 zdjęć w formacie .JPG?

W przypadku Q1 oraz Q3 Angular lepiej poradził sobie z przetwarzaniem wszystkich grup danych, zatem na pytania można odpowiedzieć twierdząco. Dla Q2 pomimo przewagi Reacta w kwestii usuwania 100 rekordów na dwóch przeglądarkach, w pozostałych przypadkach dominował Angular osiągając lepsze czasy, zatem odpowiedź na pytanie jest negatywna.

Podsumowując powyższy artykuł, warto pamiętać o fakcie, iż przeprowadzone testy wydajnościowe miały specyficzny charakter w postaci sprawdzenia wydajności renderowania w ekstremalnych warunkach. Rekomendacjami do dalszych badań mogłyby być:

- wykonanie badań z wykorzystaniem dodatkowego szkieletu,
- wykonanie badań na większej liczbie rekordów oraz zdjęć, pobierając dane pojedynczo, a następnie renderując za jednym razem po zakończeniu zapytań do serwera.

Literatura

[1] K. Bielak, B. Borek, M. Plechawska-Wójcik, Web application performance analysis using Angular, React and Vue.js frameworks, Journal of Computer Sciences Institute 23 (2022) 77-83, https://doi.org/10.35784/jcsi.2827.

[2] E. Petukhova, Sitecore JavaScript Services Framework Comparison, praca magisterska, Åbo Akademi University, Turku 2019.

[3] C. L. Mariano, Benchmarking JavaScript Frameworks, praca magisterska, Technological University Dublin, Dublin 2017.

[4] J. Kalinowska, B. Pańczyk, Comparison of tools for creating SPA applications using the examples of Angular2 and React, Journal of Computer Sciences Institute 10 (2019) 1-4, https://doi.org/10.35784/jcsi.183.

[5] R. Ferguson, JavaScript and Application Frameworks: Angular. In: Beginning JavaScript, Apress, Berkeley, 2019.

[6] E. Wohlgethan, Supporting web development decisions by comparing three major javascript frameworks: Angular, react and vue.Js, praca licencjacka, Hamburg University of applied sciences, Hamburg 2018.

[7] B. Grynhaus, J. Hudgens, R.Hunte, M. Morgan, W. Stefanovski, TypeScript na warsztacie. Praktyczny przewodnik pisania efektywnego kodu, Helion, 2022.

[8] E. Molin, Comparison of single-page application frameworks. A method of how to compare Single-Page Application frameworks written in JavaScript, praca magisterska, KTH Royal Institute of Technology, Stockholm 2016.

[9] S. Mousavi, Maintainability Evaluation of Single Page Application Frameworks : Angular2 vs. React, praca licencjacka, Linnaeus University, Kalmar 2017.

[10] Y. Fain, A. Moiseev, Angular. Programowanie z użyciem języka TypeScript, Wydanie II, Helion, 2019.

[11] D. B. Duldulao, Practical Enterprise React: Become an Effective React Developer in Your Team, Apress, 2021.

[12] C. Gackenheimer, Introduction to React, Apress, Berkeley, 2015.

[13] A. Fedosejev, React. js essentials, Packt Publishing Ltd, Birmingham, 2015.

[14] J. Voutilainen, Evaluation of Front-end JavaScript Frameworks for Master Data Management Application Development, praca licencjacka, Metropolia University of Applied Sciences, Helsinki 2017.

[15] S. Retzius, E. Sundholm, Development of an evaluation model for client-side JavaScript Frameworks, praca magisterska, Linköping University, Linköping 2022.

[16] A. Lassen, JavaScript Frameworks A qualitative evaluation and comparison of the dominant factors in Angular and React, praca magisterska, Roskilde University, Roskilde 2020.

[17] S. D. Ankush, XSS attack prevention using DOM based filtering API, praca magisterska, National Institute of Technology Rourkela, Rourkela 2014.

[18] X. Lin, P. Zavarsky, R. Ruhl, D. Linskog, Threat modeling for CSRF attacks, International Conference on Computational Science and Engineering 3 (2009) 486-491, https://doi.org/10.1109/CSE.2009.372.

[19] L. K. Shar, H. B. K. Tan, Defeating SQL injection, Computer 46(3) (2012) 69-77, https://doi.org/10.1109/MC.2012.283.

[20] A. Bhaskar, A. E.Manjunath, An Interpretation and Anatomization of Angular: A Google Web Framework, International Research Journal of Engineering and Technology (IRJET) 7(05) (2020) 7613-7619.

[21] K. Simkhada, Transitioning Angular 2 User Interface (UI) intoReact, praca licencjacka, Metropolia University of Applied Sciences, Helsinki 2017.

[22] E. Saks, JavaScript Frameworks: Angular vs React vs Vue, praca licencjacka, Haaga-Helia University of Applied Sciences, Helsinki 2019.

[23] A. Mardan, React Quickly: Painless web apps with React, JSX, Redux, and GraphQL, Manning, 2017.

[24] C. E. d. Oliveira, G. L. Turnquist, A. Antonov, Developing Java Applications with Spring and Spring Boot: Practical Spring and Spring Boot Solutions for Building Effective Applications, Packt Publishing, Wielka Brytania, 2018.

[25] Y. Li, S. Manoharan, A performance comparison of SQL and NoSQL databases, 2013 IEEE Pacific Rim Conference on Communications, Computers and Signal Processing (PACRIM), IEEE 8-13 (2013) 15-19, https://doi.org/10.1109/PACRIM.2013.6625441.

[26] R. Richards, Representational State Transfer (REST). In: Pro PHP XML and Web Services, Apress, Berkeley, 2006.

[27] S. M. Karadimitriou, E. Marshall, C. Knox, Mann-whitney u test, Sheffield Hallam University, Sheffield, 2018.

[28] T. J. Cleophas, A. H. Zwinderman, Non-parametric tests for three or more samples (Friedman and Kruskal-Wallis). Clinical data analysis on a pocket calculator: understanding the scientific methods of statistical reasoning and hypothesis testing, Springer (2016) 193-197, http://dx.doi.org/10.1007/978-3-319-27104-0_34.

[29] P. Mishra, U. Singh, C. M. Pandey,P. Mishra, G. Pandey, Application of student's t-test, analysis of variance, and covariance, Annals of cardiac anaesthesia, PUBMED 22(4) (2019) 407-411, https://doi.org/10.4103/aca.aca_94_19.

[30] Ankieta z 2022 r. wykonana przez StateOfJs, dotycząca szkieletów programistycznych używanych do tworzenia interfejsu użytkownika, https://2022.stateofjs.com/en-US/libraries/front-end-frameworks, [29.05.2023].