JOURNAL
COMPUTER SCIENCES INSTITUTE

JCSI 15 (2020) 139-142
Received: 3 November 2019
Accepted: 12 November 2019

Modern technologies for creating graphical user interface in web applications

Nowoczesne technologie tworzenia graficznego interfejsu użytkownika w aplikacjach internetowych

Mateusz Kaproń*, Beata Pańczyk

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

Abstract
The goal of the article is to compare the three most commonly used tools for creating a graphical user interface in web applications. The analysis was carried out for the currently most popular tools: Angular, React and Vue. Test applications with identical user interface, implemented in three technologies, were used for the research. The article compares, above all, the performance related to page loading time and memory usage..

Keywords: Angular; React; Vue; web application

Streszczenie
Celem artykułu jest porównanie trzech najczęściej stosowanych narzędzi do tworzenia graficznego interfejsu użytkownika w aplikacjach internetowych. Analiza została przeprowadzona dla najbardziej popularnych obecnie narzędzi: Angular, React i Vue. Do badań wykorzystano aplikacje testowe z identycznym interfejsem użytkownika, zaimplementowane w trzech technologiach. W artykule porównano przede wszystkim wydajność związaną z czasem ładowania stron oraz zajętość pamięci.

Słowa kluczowe: Angular; React; Vue; aplikacje internetowe

*Corresponding author
Email address: **mateusz.kapron@pollub.edu.pl** (M. Kaproń)

1. Wstęp

Liczba narzędzi do tworzenia graficznego interfejsu użytkownika i bibliotek opartych na języku JavaScript wciąż rośnie. Nauka ich wszystkich nie jest praktycznie możliwa. W niniejszym artykule rozpatrywane będą trzy najpopularniejsze narzędzia, wskazane jako najczęściej stosowane w 2019 roku dla aplikacji webowych typu SPA (ang. Single Page Application) [1, 2]. Architektura typu SPA [3] pozwala na dynamiczne nadpisywanie treści, co przekłada się na szybkość działania i wydajność aplikacji.

Angular [4, 5, 6] to framework i platforma do tworzenia aplikacji typu SPA, napisany w języku TypeScript. Jest wspierany oraz rozwijany przez Google. Najnowszą i najczęściej używaną wersją jest Angular v8, wydany w maju 2019r. Wśród firm korzystających z tego narzędzia znajduje się: Google, Microsoft, JPMorgan, Time Warner Cable, McDonald's, UPS, Apple i wiele innych.

React [7, 8, 9] to biblioteka JavaScript stworzona przez deweloperów Facebooka. Biblioteka jest opisywana, jako „narzędzie JavaScript do budowania interfejsów użytkownika". React wykorzystuje wirtualny DOM (ang. Document Object Model) i przechowuje cały model dokumentu w pamięci. Po zmianie stanu wyszukuje różnice między wirtualnym a rzeczywistym DOM, i w miare potrzeb y dokonuje aktualizacji.

Vue.js [10, 11, 12] mimi iż jest najmłodszą, progresywną biblioteką języka JavaScript, staje się powoli silnym konkurentem wobec dwóch poprzednich technologii.

2. Cel i tezy badawcze

Celem badań jest porównanie najbardziej popularnych technologii tworzenia interfejsu graficznego aplikacji internetowych na przykładzie Angular, React i Vue. Podstawowym kryterium porównania jest wydajność związana z czasem ładowania strony oraz obciążenie pamięci w przeglądarce. Analizowane są również inne cechy badanych narzędzi takie jak: wybrane metryki kodu, dostępność dokumentacji, trudność wytwarzania oprogramowania dla początkującego programisty aplikacji typu SPA (ang. Single Page Aplication).

Postawiono następujące tezy badawcze:
Wydajność interfejsu graficznego tworzonego w Vue jest porównywalna z analogicznym GUI dla React i Angular.
Angular jest najbardziej skomplikowanym narzędziem dla początkującego programisty aplikacji typu SPA w porównaniu do React i Vue.
Vue jest najbardziej przyjazną technologią dla początkującego programisty aplikacji typu SPA w porównaniu do React i Angular.

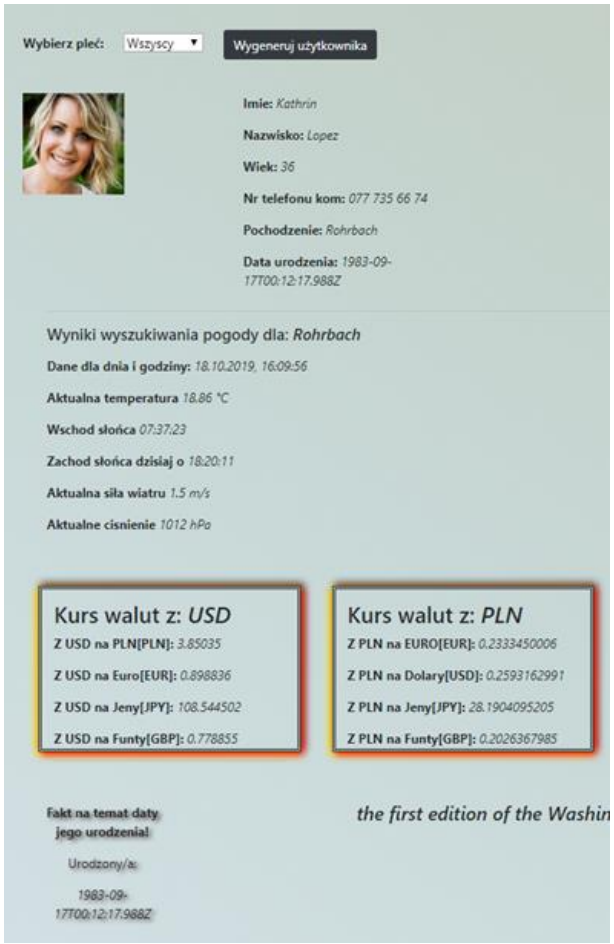
3. Metoda badań

Analizę wydajności czasowej i obciążenia pamięci wykonano za pomocą aplikacji testowej, dla której zaimplementowano trzy różne technologie generowania widoków. Wszystkie aplikacje korzystają z danych

udostępnianych przez trzy zewnętrzne API (ang. Application programming interface).

3.1. Aplikacja testowa

Aplikacja testowa generuje losowo pojedynczego użytkownika (lub wskazaną liczbę użytkowników w oparciu o zadane kryteria wyboru). W przypadku jednego użytkownika, na podstawie jego miejsca zamieszkania, aplikacja wyświetla dodatkowe dane o aktualnej pogodzie w tym miejscu. Z kolei data urodzenia losowego użytkownika jest daną wejściową do generowania dodatkowych ciekawostek, które wtedy miały miejsce. Rysunek 1 przedstawia przykładowy widok strony generującej dodatkowe informacje o losowym użytkowniku.



Rysunek 1: Interfejs graficzny aplikacji testowej

W przypadku wyboru większej liczby użytkowników (do 500), aplikacja umożliwią podanie kryterium wyboru płci i lokalizacji użytkowników. Taka funkcjonalność aplikacji pozwoliła zbadać wydajność interfejsu graficznego w oparciu o przygotowane scenariusze testowe.

3.2. Kryteria i scenariusze badawcze

Wybrane technologie zbadano według następujących kryteriów:

- czas ładowania strony w przeglądarce internetowej z czyszczeniem i bez czyszczenia pamięci przeglądarki;
- wykorzystywanie pamięci RAM;
- inne cechy technologii, takie jak dostępność dokumentacji i przykładów, wybrane metryki kodu.

W celu przeprowadzenia badań wydajności opracowano 6 scenariuszy testowych:

- S1 - generowanie pojedynczego użytkownika wraz z dodatkową informacją o pogodzie, walucie oraz losową historią wybraną na podstawie daty urodzenia;
- S2 - generowanie jednego użytkownika,
- S3 - generowanie 500 użytkowników;
- S4 - generowanie 500 mężczyzn;
- S5 - generowanie 500 mężczyzn z Australii;
- S6 - generowanie 500 kobiet z Brazylii.

Dla każdego scenariusza zostały wyznaczone:

- czasy ładowania strony,
- obciążenie pamięci.

4. Wyniki badań

Badania wykonano na komputerze Lenovo y50 z systemem operacyjnym Windows 10, o następujących podzespołach:

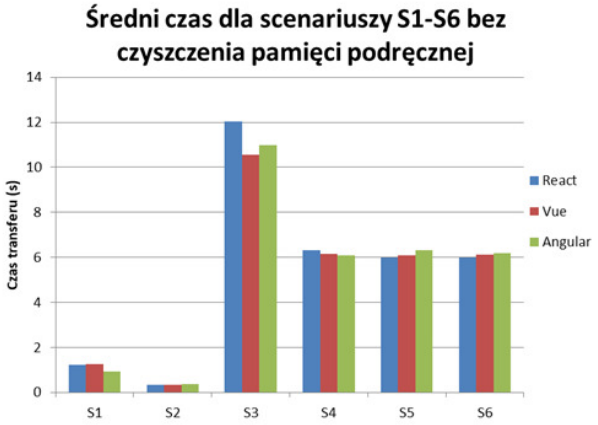
- Procesor Intel(R) Core(TM) i7-4720HQ CPU @ 2.60GHz,
- Dysk twardy SSD o pojemności 440GB,
- Pamięcią RAM 16GB DDR3 1600MHz,
- Szybkość łącza internetowego dla pobierania wynosi 100Mb/s oraz wysyłania 12Mb/s.

Czas ładowania stron oraz obciążenie pamięci dla wszystkich aplikacji testowych określono za pomocą narzędzia deweloperskiego wbudowanego w przeglądarkę Firefox Developer Edition (wersja 71.0 beta). Wszystkie pomiary powtarzano po 5 razy.

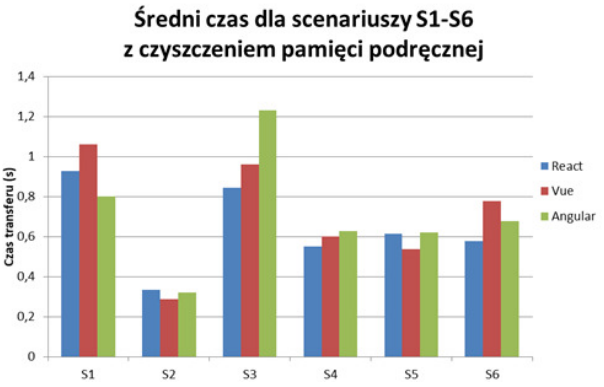
4.1. Testy wydajnościowe

Porównanie średnich czasów ładowania strony dla wszystkich scenariuszy testowych przedstawiają rysunki 2 i 3.

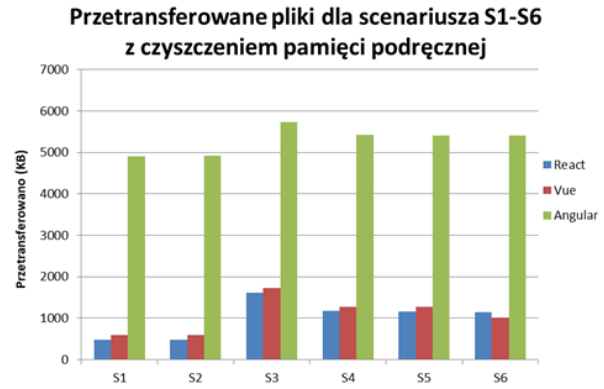
Z kolei rysunki 4 i 5 prezentują średnie obciążenie pamięci dla wszystkich scenariuszy i technologii.



Rysunek 2: Średni czas dla scenariuszy S1-S6, bez czyszczenia pamięci podręcznej przeglądarki

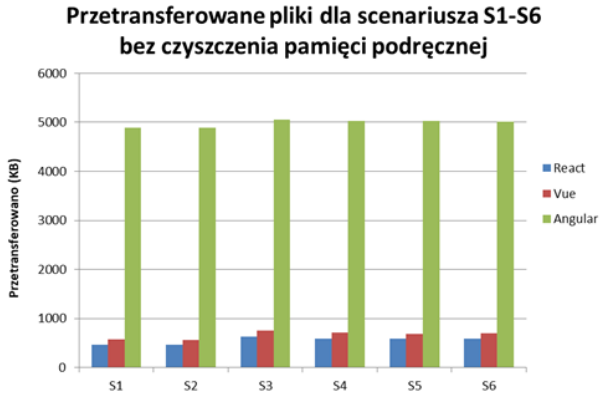


Rysunek 3: Czasy dla scenariuszy S1-S6 z czyszczeniem pamięci podręcznej przeglądarki



Rysunek 4: Przetransferowane pliki dla scenariusza S1-S6 z czyszczeniem pamięci podręcznej

Analizując szybkość ładowania strony bez czyszczenia pamięci cache w przeglądarce (rys. 2), dla aplikacji typu SPA w przypadku scenariuszy S2, S5 i S6 najszybszy okazał się React, dla scenariuszy S1 i S4 jest to Angular, a dla S3 – Vue. W przypadku, gdy pamięć przeglądarki jest czyszczona (rys. 3) – najwydajniejszym dla S3, S4 i S6 jest React, dla S1 – Angular, a Vue dla S2 i S5. Podsumowując wyniki badania wydajności czasowej można stwierdzić, że żadna z technologii nie jest znacząco lepsza w stosunku do pozostałych.



Rysunek 5: Przetransferowane pliki dla scenariusza S1-S6 bez czyszczenia pamięci podręcznej

Analizując liczbę transferowanych danych [KB] bez czyszczenia pamięci cache w przeglądarce (rys. 5), w przypadku wszystkich scenariuszy najbardziej efektywny jest React, jednak Vue zużywa niewiele więcej pamięci. W przypadku, gdy pamięć przeglądarki jest czyszczona (rys. 4) – wyniki są bardzo zbliżone do tych, gdy cache przeglądarki nie był czyszczony. W przypadku tego kryterium – Angular wypada dużo gorzej od pozostałych dwóch technologii.

4.2. Dostępność dokumentacji

Każde z omawianych narzędzi posiada rozbudowaną dokumentację, która jest stale rozwijana i aktualizowana. Zarówno Angular [6], React [9], jak i Vue [12] posiadają bardzo obszerne dokumentacje, a ich twórcy starają się w łatwy sposób przedstawić sposoby ich wykorzystania. Dla każdej technologii, poza darmową dokumentacją, kursem, tutorialem, istnieje również możliwość nauki poprzez inne płatne kursy online. Warto zwrócić uwagę, że większość kursów dostępnych w sieci jest w języku angielskim. W pracy do porównania dostępności kursów skorzystano z danych portalu [13]. Liczba kursów dla Angular wynosi 599, dla React - 570 a dla Vue tylko 216. Wynika to z faktu, że jest to technologia najnowsza, która dopiero zdobywa popularność wśród programistów front-end.

4.3. Wybrane metryki kodu

Korzystając z aplikacji testowych, w tabeli 1 przedstawiono porównanie metryk kodu i innych mierzalnych wartości istotnych dla omawianych narzędzi.

Tabela 1: Metryki kodu i inne wartości mierzalne dla aplikacji testowych

Narzędzie	React	Vue	Angular
Zajmowana pamięć [MB]	154	156	273
Liczba plików	29 325	18 128	31 011
Liczba folderów	4188	3023	3909
Liczba komponentów	6	5	3
Liczba wierszy kodu	397	337	263

4.4. Wybrane metryki kodu

Subiektywna ocena niemierzalnych cech technologii widoków została zestawiona w tabeli 2. Każde kryterium oceniono w skali od 1 do 10, gdzie większa liczba oznacza lepszą ocenę.

Tabela 2: Oceny różnych właściwości badanych technologii (zielonym kolorem są zaznaczone najwyższe oceny, czerwonym – najniższe)

Narzędzie	React	Vue	Angular
Metryki kodu	8	8	7
Dokumentacja	9	9	9
Wydajność	8	7	7
Próg wejścia	6	6	2
Czas implementacji	6	5	3
Sumaryczna ocena	37	35	26

React i Vue uzyskały zbliżone wyniki w ostatecznej ocenie punktowej. Angular uzyskał mniej punktów ze względu na wysoki próg wejścia oraz czas implementa-

cji aplikacji testowej. Porównując metryki kodu, aplikacja w Angular ma dużo więcej plików i zajmuje dużo więcej pamięci w porównaniu do React i Vue. Dokumentacja dostępna dla każdego narzędzia jest na podobnym poziomie. Z tabeli 2 wynika, że pod względem metryk kodu, wydajności, progu wejścia, czasu implementacji React i Vue są bardzo zbliżone w ocenie.

5. Wnioski

Rozważane w niniejszym artykule technologie są najczęściej stosowanymi narzędziami do tworzenia aplikacji webowych typu SPA. Podsumowując rezultaty badań przedstawione w rozdziale 4, oraz mając na uwadze doświadczenie własne autorów, można sformułować następujące wnioski:

- Angular jest dobrym narzędziem dla doświadczonych programistów. Nauka tego frameworka jest trudna i wymaga dużej wiedzy wstępnej. Struktura aplikacji w Angular nadaje się bardzo dobrze w przypadku rozbudowanych serwisów.
- Struktura aplikacji w React i Vue jest lepsza w przypadku prostych i mniej rozbudowanych serwisów.
- Vue jest technologią, której można się nauczyć najszybciej, dlatego też warto ją polecić szczególnie początkującym programistom front-end.
- czasy ładowania stron dla wszystkich technologii z czyszczeniem i bez czyszczenia pamięci podręcznej nie wykazują istotnych różnic (Rys. 2 i 3), ale stosunkowo najbardziej wydajny jest React;
- obciążenie pamięci jest największe w przypadku Angular. React i Vue – wykazują porównywalne wielkości (Rys. 4 i 5).

Powyższe wnioski dają podstawę stwierdzeniu, że postawione na wstępie tezy badań zostały potwierdzone.

Literatura

[1] Najpopularniejsze technologie JavaScript, https://2018.stateofjs.com, [22.05.2019]

[2] Serwis społecznościowy dotyczący tematyki wytwarzania oprogramowania https://stackoverflow.com, [22.10.2019].

[3] S. Michael: Single Page Web Applications Programowanie aplikacji internetowych z JavaScript. HELION, Gliwice 2015.

[4] F. Yakov.: Angular2 programowanie z użyciem języka TypeScript. HELION, Gliwice 2018.

[5] G. Kunz., Angular2 tworzenie interaktywnych aplikacji internetowych. HELION, Gliwice 2017.

[6] Dokumentacja Angular, https://angular.io/docs, [26.10.2019].

[7] Freeman A., Pro React 16, Apress, 2019.

[8] A. Cassio de Sousa, React dla zaawansowanych. Helion, Gliwice 2017.

[9] Dokumentacja React, https://reactjs.org/docs/getting-started.html, [28.10.2019].

[10] O. Fillipova, Vue.js 2 Tworzenie reaktywnych aplikacji WWW. Helion, Gliwice 2017.

[11] A. Freeman, Pro Vue. js 2. Apress, 2018.

[12] Dokumentacja Vue, https://vuejs.org/v2/guide/, [26.10.2019]

[13] Kursy online, https://www.udemy.com/, [23.10.2019]