|  |  |
| --- | --- |
| **Politechnika Świętokrzyska w Kielcach**  **Wydział Elektrotechniki, Automatyki i Informatyki** | |
| **Technologie obiektowe - projekt** | |
| Etap 7 | |
| **Mateusz Bonar**  **Wiktor Wójcik**  **Damian Kozakowski**  **1ID21A** |  |

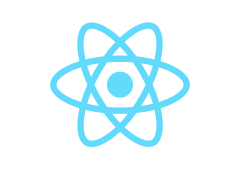
1. **Wstęp oraz zadanie projektowe**

Naszym zadaniem projektowym było przygotowanie temtu numer 34 który poruszał temat:

**Porównanie wybranych frameworków - porównamy między sobą rożne frameworki: React.js, Angular, Vue.js , porównamy wydajności tych technologii wraz   
z wykorzystaniem aplikacji z częścią back-end projektu.**

Przed przystąpieniem do przeprowadzonych przez nas testów oraz wyciągniętych wniosków przybliżmy trochę poszczególne frameworki na które zostały wykorzystane do naszego projektu.

1. **Wstęp teoretyczny na temat wykorzystanych technologii.**
   1. **React.js**



Rys. **Błąd! W dokumencie nie ma tekstu o podanym stylu.**.1 Logo biblioteki React.js

**React.js** jest to biblioteka języka programowania JavaScript, która wykorzystywana jest do tworzenia interfejsów graficznych aplikacji internetowych. Została stworzona przez Jordana Walke, programistę Facebooka, a zainspirowana przez rozszerzenie języka PHP - XHP. React.js pozwala na opracowywanie aplikacji w technologii SPA (ang. *Single Page Aplication*). Schematem pracy z wykorzystaniem tej biblioteki jest podział elementów na komponenty które może renderować do określonego elementu w DOM (ang. *Document Object Model*) przy użyciu biblioteki React DOM.

Element, który wyróżnia ta bibliotekę na płaszczyźnie wielu innych dostępnych narzędzi do tworzenia internetowych graficznych interfejsów jest wykorzystanie wirtualnego DOMu. React tworzy pamięć podręczną struktury danych w pamięci. Oblicza wynikające z niej różnice, a następnie wydajnie aktualizuje wyświetlany DOM przeglądarki. Pozwala to programiście pisać kod tak jakby cała strona była renderowana przy każdej zmianie, podczas gdy biblioteki React wyświetlają tylko te elementy, które faktycznie się zmieniają.



Rys. **Błąd! W dokumencie nie ma tekstu o podanym stylu.**.2 Rysunek przedstawiający zasadę wykorzystania Virtual DOM

React wykorzystuje również JSX czuli rozszerzenie składni języka JavaScript. Podobnie do HTML, JSX zapewnia sposób strukturyzacji renderowania komponentów przy użyciu składni znanej wielu programistom.

* 1. **Angular 9**



Rys.**Błąd! W dokumencie nie ma tekstu o podanym stylu.**.3 Logo Angular

Kolejnym wykorzystanym przez nas frameworkiem był Angular w wersji 9. Angular to rozwiązanie oparte na języku TypeScript, kierowana przez Angular Team w Google. Angular jest już nie biblioteką do tworzenia graficznych interfejsów użytkownika lecz narzędziem zapewniającym w swym działaniu wiele bibliotek, które niestety w porównaniu do React.js trzeba uzupełniać poprzez npm. Wykorzystywana wersja 9 tego oprogramowania przenosi wszystkie aplikacje do domyślnego korzystania z kompilatora i środowiska Ivy.

Angular ma hierarchiczny zastrzyk zależności, znacznie lepszy niż AngularJS, w którym klasy nie są od siebie zależne. Zamiast tego zwracają się w stronę źródeł zewnętrznych, które zapewniają wyższą wydajność aplikacji mobilnych Angular. Oferuje on gotowe elementy do projektowania materiałów w elementach nawigacyjnych, kontrolkach formularzy, oknach pop-up, układach i tabelach danych. Kolejnym elementem wyróżniającym Angular na tle React.js jest wykorzystanie dwukierunkowego powiązania danych zapewnia to, że ​​stan modelu zmienia się automatycznie po każdej zmianie elementu interfejsu użytkownika i odwrotnie.

* 1. **Vue.js**



Rys.**Błąd! W dokumencie nie ma tekstu o podanym stylu.**.4 Logo Vue.js

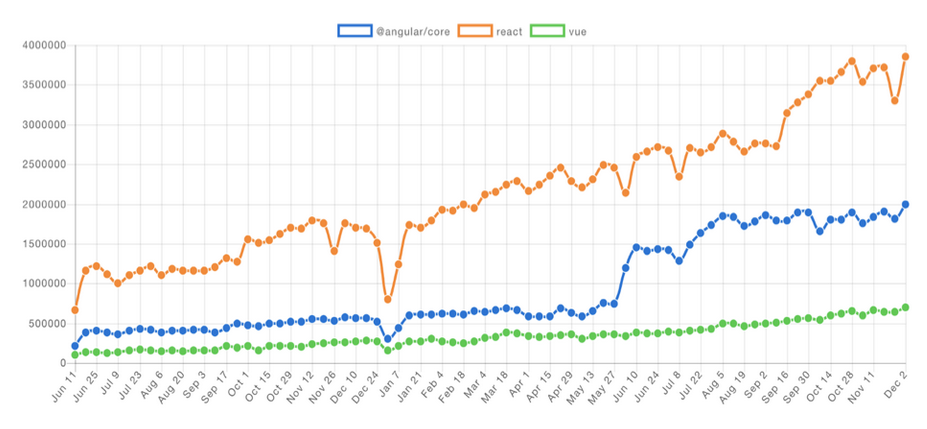
Na potrzeby projektu wykorzystaliśmy również do porównania framework Vue.js, który również jest wykorzystywany do budowania interfejsów użytkownika. Został stworzony przez Evana You   
i jest utrzymywany przez niego i resztę aktywnych członków podstawowego zespołu z różnych firm, takich jak Netify i Netguru. Rozwiązanie Evana You miało na celu wyodrębnienie lubianych   
i efektywnych elementow z Angular celem zbudowania „lżejszego” odpowiednika.

Komponenty Vue rozszerzają podstawowe elementy HTML o enkapsulację kodu wielokrotnego użytku. Na wysokim poziomie komponenty są niestandardowymi elementami, do których kompilator Vue przywiązuje zachowanie. W Vue komponent jest zasadniczo instancją Vue ze wstępnie zdefiniowanymi opcjami. Vue używa składni szablonu opartej na HTML , która pozwala powiązać renderowany DOM z danymi bazowej instancji Vue. Wszystkie szablony Vue są prawidłowym kodem HTML, który można analizować za pomocą przeglądarek zgodnych ze specyfikacją i analizatorów składni HTML. Posiada również system reaktywności, który wykorzystuje proste obiekty JavaScript i zoptymalizowane na ponowne renderowanie. Każdy komponent śledzi swoje zależności reaktywne podczas renderowania, dzięki czemu system dokładnie wie, kiedy wykonać ponowne renderowanie i które elementy ponownie renderować, wykorzystanie podobniej zależności widzimy zarówno w React.js.

**Dlaczego porównanie Vue.js, Angular 2+ oraz React.js?**

Na przestrzeni tworzonych przez nas projektów zawsze stawaliśmy nad wyborem podczas przygotowywania części front-end naszych projektów. Najczęstszymi wyborami polecanymi były właśnie te trzy frameworki. Podczas tego projektu chcieliśmy się zagłębić w poszczególne rozwiązania celem zauważenia różnic oraz wydajności na poszczególnych frameworkach. Wybrane przez nas narzędzia znajdują się w czołówce rozwiązań do graficznego tworzenia interfejsów aplikacji internetowych.

Według danych na 2019 rok. Najczęściej używanym językiem programowania jest JavaScript a używa go ponad 80% badanych. W tym React i Angular mają prawie taki sam poziom współczynnika używalności (user ratio) w kategorii popularnych frameworków front-end.



Rys .**Błąd! W dokumencie nie ma tekstu o podanym stylu.**.5 Statystyki wyszukiwania na stackowerflow.com frameworków

1. **Słowem wstępu o porównaniu frameworków.**

Podczas porównywania rozwiązań skupiliśmy się na szybkości załadowania elementów oraz wyrenderowania strony.

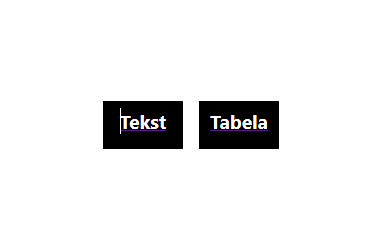
* Czas uzyskania odpowiedzi z REST API wielkości danych np. 1kB, 3kB, 5 kB, 10kB, 20kB, 50kB, 1MB, 3 MB, 5 MB, 10 MB.
* Czas załadowania statycznego tekstu z pliku JSON.
* Czas załadowania danych statycznych (Tabela) z pliku JSON.
* Czas wykonania operacji CRUD( *Create Read Update Delete*) poszczególnych frameworków.

Do uzyskania dokładniejszych pomiarów oraz automatyzacji wykorzystamy bibliotekę Selenium.

Przygotowaliśmy również część back-end z wykorzystaniem Spring Boot.

1. **Przygotowane elementy oraz część Selenium**

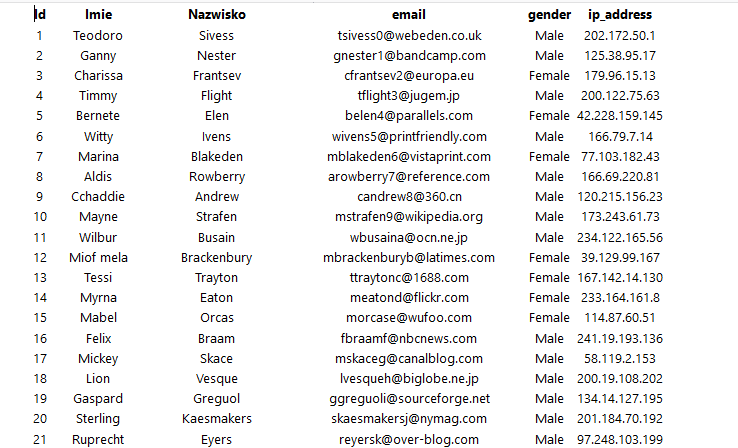
Poniżej prezentują się widoki elementów przygotowanych w każdym z trzech rozwiązań.

Poglądowo poniżej zostanie zaprezentowany widok z React.js

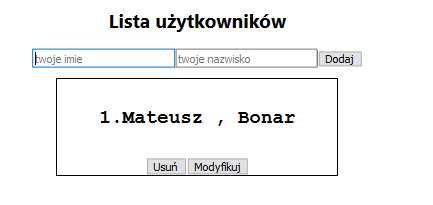
Rys.**Błąd! W dokumencie nie ma tekstu o podanym stylu.**.6 Strona główna aplikacji wraz z wyborem



Rys.**Błąd! W dokumencie nie ma tekstu o podanym stylu.**.7 Komponent prezentujący tekst statyczny



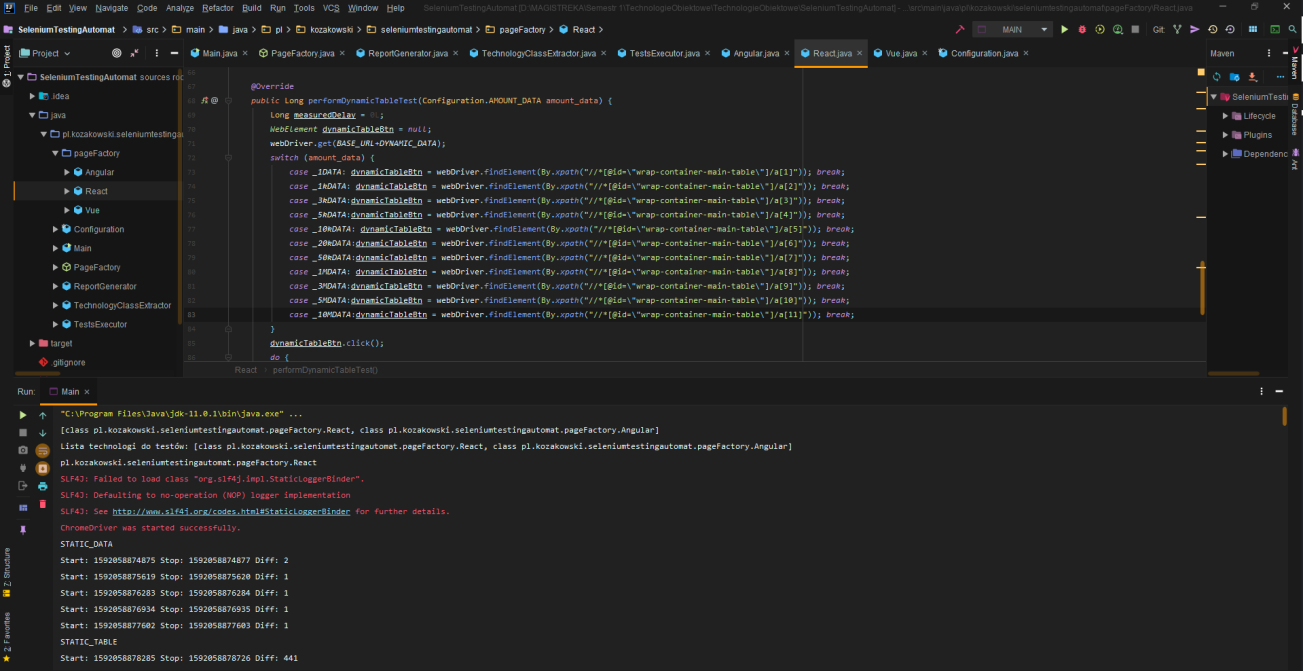
Rys **Błąd! W dokumencie nie ma tekstu o podanym stylu.**.8 Tabela statyczna



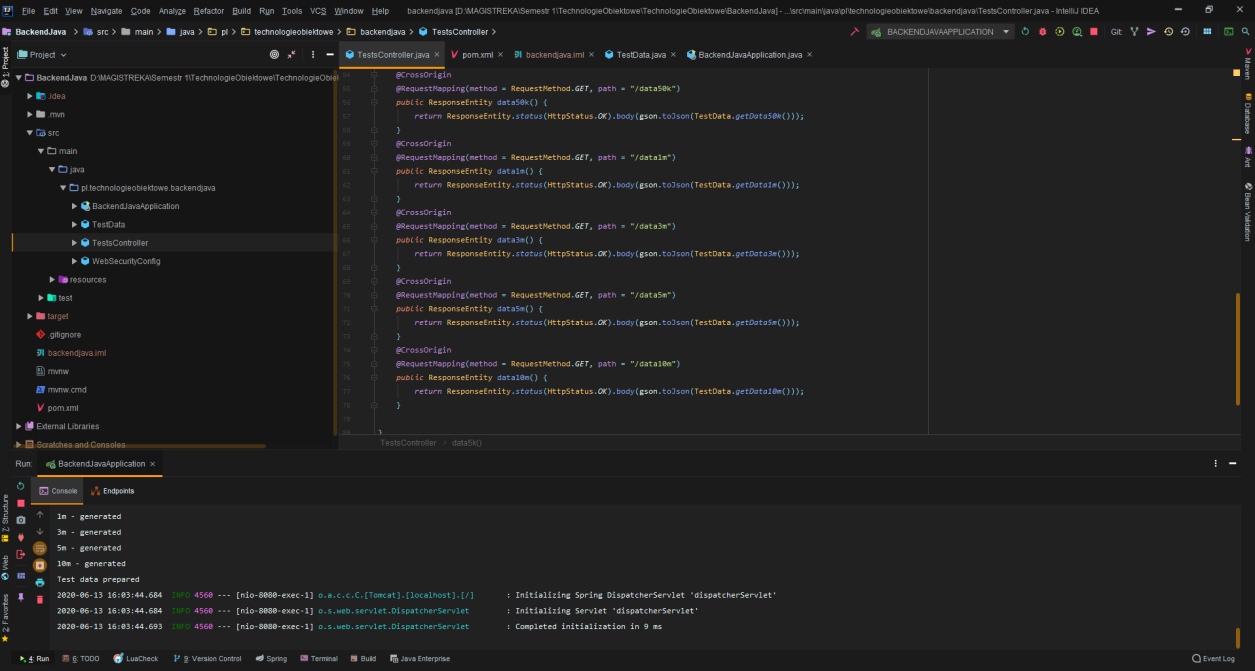
Rys **Błąd! W dokumencie nie ma tekstu o podanym stylu.**.9 Komponent odpowiadający za częśc CRUD projektu



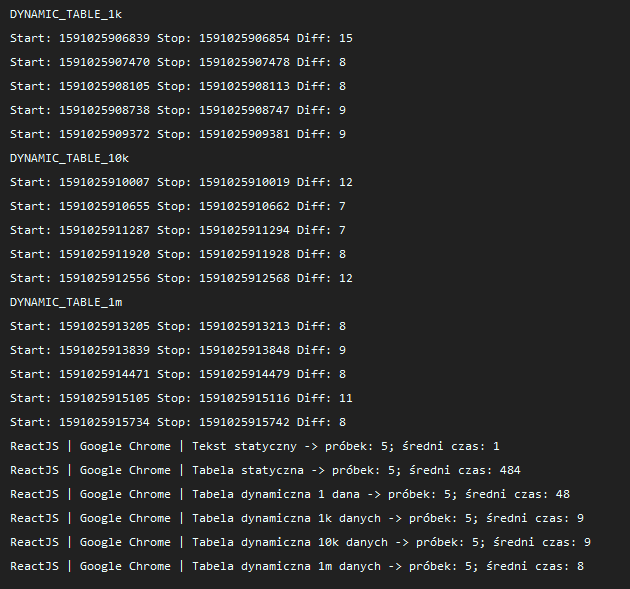
Rys **Błąd! W dokumencie nie ma tekstu o podanym stylu.**.10 Wyidok prezentujący zapytania do części back-end



Rys **Błąd! W dokumencie nie ma tekstu o podanym stylu.**.11 Prezentacja częsci projektu napisanej w selenium



Rys **Błąd! W dokumencie nie ma tekstu o podanym stylu.**.12 Prezentacja części back-end projektu przygotowaną w Spring Boot



Rys **Błąd! W dokumencie nie ma tekstu o podanym stylu.**.13Przykładowe wyniki uzyskiwane za pomocą selenium

1. **Uzyskane pomiary oraz wykresy**

**Poniżej widzimy wyniki naszych pomiarów:**

Wykonano celem dokładniejszych pomiarów po 5 próbek:

|  |  |  |  |
| --- | --- | --- | --- |
| RODZAJE\_POMIARÓW | React.js | Angular | Vue.js |
| STATIC\_DATA | 5 | 12 | 20 |
| STATIC\_DATA | 2 | 6 | 1 |
| STATIC\_DATA | 2 | 16 | 9 |
| STATIC\_DATA | 4 | 7 | 1 |
| STATIC\_DATA | 3 | 10 | 1 |
|  |  |  |  |
| STATIC\_TABLE | 557 | 2 | 197 |
| STATIC\_TABLE | 684 | 1 | 154 |
| STATIC\_TABLE | 450 | 1 | 220 |
| STATIC\_TABLE | 451 | 1 | 200 |
| STATIC\_TABLE | 466 | 1 | 200 |
|  |  |  |  |
| DYNAMIC\_TABLE | 20 | 11 | 4 |
| DYNAMIC\_TABLE | 11 | 10 | 8 |
| DYNAMIC\_TABLE | 9 | 9 | 5 |
| DYNAMIC\_TABLE | 7 | 21 | 4 |
| DYNAMIC\_TABLE | 13 | 10 | 4 |
|  |  |  |  |
| DYNAMIC\_TABLE\_1k | 16 | 14 | 4 |
| DYNAMIC\_TABLE\_1k | 16 | 5 | 4 |
| DYNAMIC\_TABLE\_1k | 8 | 18 | 11 |
| DYNAMIC\_TABLE\_1k | 9 | 5 | 4 |
| DYNAMIC\_TABLE\_1k | 15 | 23 | 3 |
|  |  |  |  |
| DYNAMIC\_TABLE\_3k | 8 | 6 | 2 |
| DYNAMIC\_TABLE\_3k | 7 | 17 | 9 |
| DYNAMIC\_TABLE\_3k | 7 | 7 | 4 |
| DYNAMIC\_TABLE\_3k | 11 | 6 | 4 |
| DYNAMIC\_TABLE\_3k | 11 | 9 | 4 |
|  |  |  |  |
| DYNAMIC\_TABLE\_5k | 8 | 6 | 12 |
| DYNAMIC\_TABLE\_5k | 8 | 14 | 4 |
| DYNAMIC\_TABLE\_5k | 10 | 5 | 4 |
| DYNAMIC\_TABLE\_5k | 8 | 18 | 5 |
| DYNAMIC\_TABLE\_5k | 9 | 6 | 8 |
|  |  |  |  |
| DYNAMIC\_TABLE\_10k | 6 | 5 | 4 |
| DYNAMIC\_TABLE\_10k | 10 | 19 | 4 |
| DYNAMIC\_TABLE\_10k | 13 | 6 | 4 |
| DYNAMIC\_TABLE\_10k | 7 | 13 | 9 |
| DYNAMIC\_TABLE\_10k | 9 | 5 | 4 |
|  |  |  |  |
| DYNAMIC\_TABLE\_20k | 11 | 7 | 6 |
| DYNAMIC\_TABLE\_20k | 10 | 7 | 5 |
| DYNAMIC\_TABLE\_20k | 10 | 10 | 12 |
| DYNAMIC\_TABLE\_20k | 10 | 6 | 4 |
| DYNAMIC\_TABLE\_20k | 13 | 21 | 4 |
|  |  |  |  |
| DYNAMIC\_TABLE\_50k | 13 | 7 | 4 |
| DYNAMIC\_TABLE\_50k | 12 | 19 | 15 |
| DYNAMIC\_TABLE\_50k | 10 | 9 | 5 |
| DYNAMIC\_TABLE\_50k | 9 | 8 | 7 |
| DYNAMIC\_TABLE\_50k | 11 | 7 | 11 |
|  |  |  |  |
| DYNAMIC\_TABLE\_1m | 34 | 35 | 32 |
| DYNAMIC\_TABLE\_1m | 36 | 25 | 31 |
| DYNAMIC\_TABLE\_1m | 34 | 30 | 30 |
| DYNAMIC\_TABLE\_1m | 35 | 29 | 31 |
| DYNAMIC\_TABLE\_1m | 37 | 34 | 29 |
|  |  |  |  |
| DYNAMIC\_TABLE\_3m | 96 | 83 | 80 |
| DYNAMIC\_TABLE\_3m | 91 | 87 | 75 |
| DYNAMIC\_TABLE\_3m | 83 | 81 | 87 |
| DYNAMIC\_TABLE\_3m | 89 | 69 | 82 |
| DYNAMIC\_TABLE\_3m | 89 | 82 | 107 |
|  |  |  |  |
| DYNAMIC\_TABLE\_5m | 136 | 124 | 158 |
| DYNAMIC\_TABLE\_5m | 143 | 116 | 128 |
| DYNAMIC\_TABLE\_5m | 140 | 125 | 139 |
| DYNAMIC\_TABLE\_5m | 143 | 143 | 136 |
| DYNAMIC\_TABLE\_5m | 152 | 121 | 127 |
|  |  |  |  |
| DYNAMIC\_TABLE\_10m | 263 | 233 | 272 |
| DYNAMIC\_TABLE\_10m | 247 | 233 | 263 |
| DYNAMIC\_TABLE\_10m | 245 | 236 | 271 |
| DYNAMIC\_TABLE\_10m | 256 | 247 | 270 |
| DYNAMIC\_TABLE\_10m | 252 | 222 | 250 |

Wyciągnięte średnie z poszczególnych próbek pięciu pomiarów:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| POMIAR | LICZBA PRÓBEK | React | Angular | Vue |
| STATIC\_DATA | 5 | 3 | 10 | 10 |
| STATIC\_TABLE | 5 | 521 | 100 | 200 |
| DYNAMIC\_TABLE\_1 | 5 | 12 | 12 | 5 |
| DYNAMIC\_TABLE\_1k | 5 | 12 | 13 | 5 |
| DYNAMIC\_TABLE\_3k | 5 | 8 | 9 | 4 |
| DYNAMIC\_TABLE\_5k | 5 | 8 | 9 | 6 |
| DYNAMIC\_TABLE\_10k | 5 | 9 | 9 | 5 |
| DYNAMIC\_TABLE\_20k | 5 | 10 | 10 | 6 |
| DYNAMIC\_TABLE\_50k | 5 | 11 | 10 | 8 |
| DYNAMIC\_TABLE\_1m | 5 | 35 | 30 | 30 |
| DYNAMIC\_TABLE\_3m | 5 | 89 | 80 | 86 |
| DYNAMIC\_TABLE\_5m | 5 | 142 | 125 | 137 |
| DYNAMIC\_TABLE\_10m | 5 | 250 | 234 | 265 |

Przedstawienie danych na wykresie w zależności ms do wykonywanej operacji:

Poniżej prezentowane są również dane wykonania czynności CREATE, UPDATE oraz DELETE na poszczególnych frameworkach.

|  |  |  |  |
| --- | --- | --- | --- |
| POMIARY | LICZBA\_PROBEK | React | Angular |
| DODAWANIE | 200 | 2 | 1 |
| EDYCJA | 100 | 3 | 1 |
| USUWANIE | 200 | 1 | 1 |

Pobrano pomiary w liczbie próbek podanej w tabeli wyciągnięto średnią i przedstawiono na wykresie. Przedstawiającym zależność w ms do wykonywanej operacji.

1. **Wnioski**

*Na podstawie uzyskanych danych oraz wiedzy sporządźmy wnioski na temat frameworkow: Angular, Vue.js, React.js.*

Jak widzimy wyniki przedstawiają przewagę Angular oraz Vue nad pozostałymi narzędziami. Angular to pełnowartościowy framework do tworzenia oprogramowania nie wymagający żadnych dodatkowych bibliotek przez co te przygotowane są pod względem wydajności w lepszy sposób. Oczywiście Angular wspiera dwukierunkowe wiązanie danych co jest mniej wydajne niż w przypadku React.js opartym na jednokierunkowym wiązaniu danych, lecz na kolejnych wersjach element został tak zoptymalizowany że może dorównywać React.js. Ale na potrzeby naszego projektu tworzone aplikacje nie posiadają ogromnej złożoności więc to również miało wpływ na wyniki. Patrząc na wyniki które otrzymaliśmy wykorzystując Vue.js zauważamy ze osiągnął on wręcz świetne wyniki w porównaniu do konkurencji. Niestety mała popularność oraz walka z narzędziami wspieranymi przez wielkie korporacje sprawia ze nie jest on tak często wykorzystywany jak React.js czy Angular.

Na przykładnie CRUD możemy wywnioskować że przy dużej ilości próbek ciągłe porównanie drzewa DOM z wykorzystaniem Virtual DOM wpłynęło na uzyskanie gorszych wyników w porównaniu do Angular. Stan w React.js każdy komponent posiada swój stan, główną wadą jest to, że stan globalny musi być przechowywany w wielu różnych częściach aplikacji, a dane są ręcznie przekazywane wokół różnych poziomów drzewa komponentów. Angular wykorzystuje prawdziwy DOM, który aktualizuje całą strukturę drzewa, nawet jeśli zmiany miały miejsce w jednym elemencie. Prawdziwy DOM jest uważany za wolniejszy i mniej skuteczny niż wirtualny DOM. Aby zrekompensować tę wadę, Angular wykorzystuje wykrywanie zmian w celu identyfikacji komponentów, które należy zmienić. Dlatego prawdziwy DOM w Angular działa równie skutecznie jak wirtualny DOM w React. Przez co również w tym przypadku mamy przewagę dla Angular. Być może pojście inną drogą w projekcie i wykorzystanie innych zależności pisania aplikacji (wykorzystanie sposobu pisania aplikacji z wykorzystanie React Hooks lub Redux oraz Thunk) React.js pozwoliłoby na osiagniecie lepszych wyników.

*Na potrzeby projektu poruszmy też temat dlaczego skoro Angular oraz Vue.js są wydajniejsze to React.js jest równie popularny?*

Spowodowane jest to krzywą uczenia się danego narzędzia oraz dostępności do materiałów dodatkowych. Angular jest narzędziem posiadającym wysoki próg wejścia istnieje tutaj wiele niepotrzebnych składni a poznanie związanych z nim pojęć zajmuje więcej czasu niż w przypadku React.js. Kolejnym elementem przemawiającym na rzecz React.js jest to że mimo że TypeScript bardzo przypomina JavaScript, jego nauka zajmuje trochę czasu. Ponieważ środowisko jest stale aktualizowane, programista musi włożyć dodatkowy wysiłek w naukę. Ale posiada on zalety o których też trzeba powiedzieć czyli jak wcześniej wspominano jest on frameworkiem. Posiada zaktualizowane wersje oprogramowania aby nadążać za konkurentem. Na przykład, chociaż uważano, że React wygrał z powodu wirtualnego DOM, Angular wyrównał wynik, wdrażając wykrywanie zmian. Chociaż Angular był uważany za zwycięzcę, ponieważ został opracowany przez tak autorytatywną firmę jak Google, ogromna oddana społeczność React w pełni zrekompensowała reputację Google i upodobniła React do Angulara.

Podsumowując uzyskane przez nas wyniki przemawiają za wykorzystywaniem Vue ze względu na wydajność. Ewidetnie wyniki uzyskiwane przez ten framework są lepsze w porównaniu do konkurencji. Niestety wsparcie takiej korporacji jak Google sprawia ze to właśnie Angular jest frameworkiem najczęściej wykorzystywanym przez firmy programistyczne.