

Politechnika Świętokrzyska
Wydział Elektrotechniki, Automatyki i Informatyki
Projekt: Technologie obiektowe

Nr. projektu: 34

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

Wykonujący:

Damian Kozakowski
Mateusz Bonar
Wiktor Wójcik
Grupa: 1ID21A

Wstęp oraz zadanie projektowe

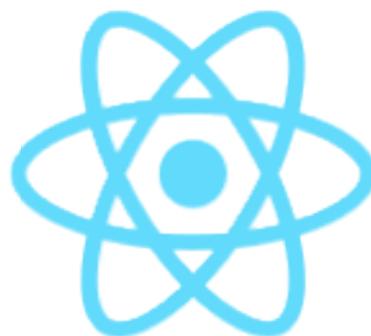
Naszym zadaniem projektowym było przygotowanie tematu numer 34 który poruszał temat:

Porównanie wybranych frameworków - porównamy między sobą różne 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.

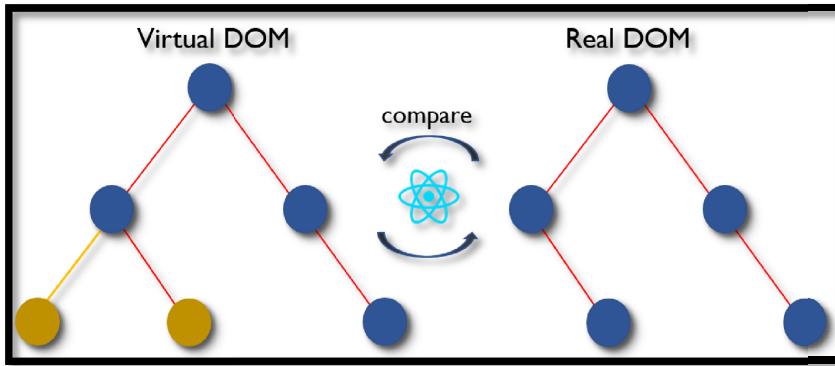
1.1. React.js



Rys.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 Application*). 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.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.

1.2. Angular 9



Rys.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.

1.3. Vue.js



Rys.4 Logo Vue.js

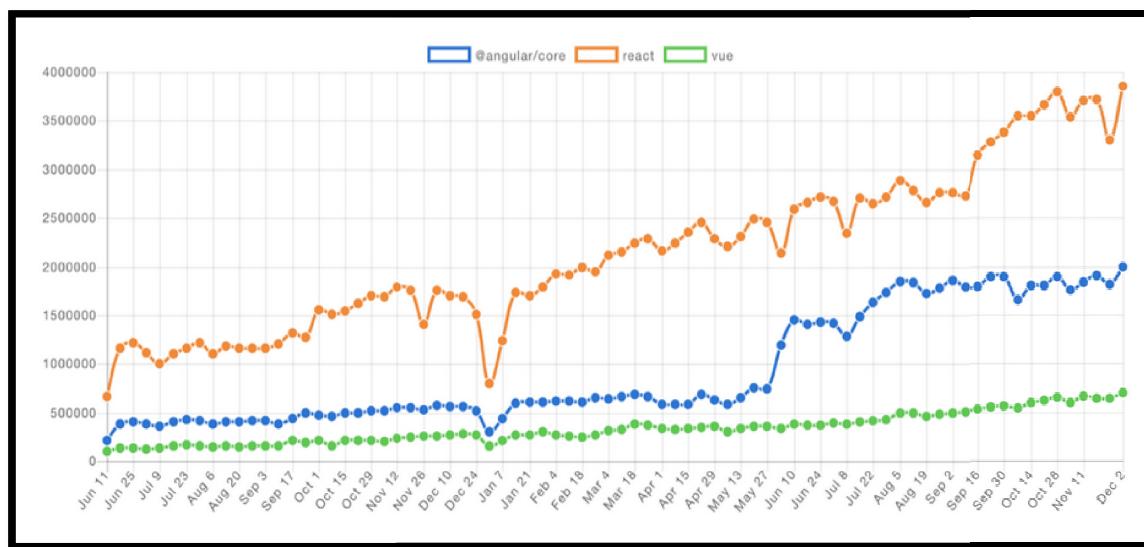
Na potrzeby projektu wykorzystaliśmy również do porównania framework Vue.js, który również jest wykorzystywany do budowania interfejsów użytkownika. Został stworzony przez Evana You i jest utrzymywany przez niego i resztę aktywnych członków podstawowego zespołu z różnych firm, takich jak Netifyi Netguru. Rozwiążanie Evana You miało na celu wyodrębnienie lubianych i efektywnych elementów 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 podobnej zależności widzimy zarówno w React.js.

2. 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. 1 Statystyki wyszukiwania na stackoverflow.com frameworków

3. Opis projektu.

Projekt jest podzielony na 5 części:

- „BackendJava” – projekt napisany w języku JAVA, z użytą biblioteką Spring dzięki której możliwa jest komunikacja REST.
- „SeleniumTestingAutomat” – projekt napisany w języku JAVA, używa biblioteki Selenium dzięki któremu możliwe jest zautomatyzowanie testów.
- „FrontReact” – projekt napisany dzięki bibliotece React.js wykorzystujący język JavaScript.
- „FrontAngular” - projekt napisany dzięki bibliotece Angular 2+ wykorzystujący język TypeScript.
- „FrontVue” - projekt napisany dzięki bibliotece Vue.js wykorzystujący język JavaScript.

4. Słówem 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.

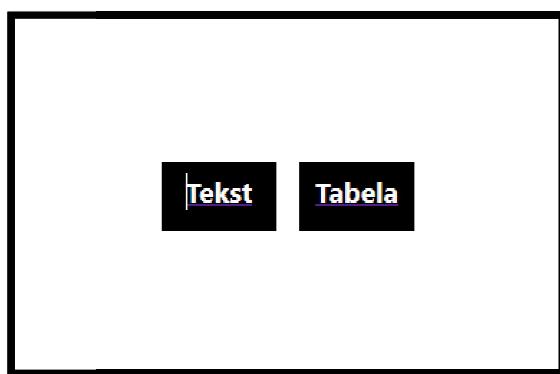
Do uzyskania dokładniejszych pomiarów oraz automatyzacji wykorzystamy bibliotekę Selenium.

Przygotowaliśmy również część back-end z wykorzystaniem Spring Boot.

5. 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.1 Strona główna aplikacji wraz z wyborem



Rys.2 Komponent prezentujacy tekst statyczny

id	Imie	Nazwisko	email	gender	ip_address
1	Teodoro	Sivess	tsivess0@webeden.co.uk	Male	202.172.50.1
2	Ganny	Nester	gnester1@bandcamp.com	Male	125.38.95.17
3	Charissa	Frantsiev	cfrantsev2@euroopa.eu	Female	179.96.15.13
4	Timmy	Flight	tflight3@jugem.jp	Male	200.122.75.63
5	Bernete	Elen	belen4@parallels.com	Female	42.228.159.145
6	Witty	Ivens	wivens5@printfriendly.com	Male	166.79.7.14
7	Marina	Blakeden	mblakeden6@vistaprint.com	Female	77.103.182.43
8	Aldis	Rowberry	arowberry7@reference.com	Male	166.69.220.81
9	Cchaddie	Andrew	candrew8@360.cn	Male	120.215.156.23
10	Mayne	Strafen	mstrafen9@wikipedia.org	Male	173.243.61.73
11	Wilbur	Busain	wbusaina@ocn.ne.jp	Male	234.122.165.56
12	Miof mela	Brackenbury	mbrackenburyb@latimes.com	Female	39.129.99.167
13	Tessi	Trayton	ttraytonc@1688.com	Female	167.142.14.130
14	Myrna	Eaton	meatond@flickr.com	Female	233.164.161.8
15	Mabel	Orcas	morcase@wufou.com	Female	114.87.60.51
16	Felix	Braam	fbraam@nbcnews.com	Male	241.19.193.136
17	Mickey	Skace	mskaceg@canalblog.com	Male	58.119.2.153
18	Lion	Vesque	lvesqueh@biglobe.ne.jp	Male	200.19.108.202
19	Gaspard	Greguol	ggreguoli@sourceforge.net	Male	134.14.127.195
20	Sterling	Kaesmakers	skaesmakersj@nymag.com	Male	201.184.70.192
21	Ruprecht	Eyers	reyersk@over-blog.com	Male	97.248.103.199

Rys.3 Tabela statyczna

Lista użytkowników

1. Mateusz , Bonar

Rys.4 Komponent odpowiadający za część CRUD projektu



Rys.5 Widok prezentujący zapytania do części back-end

```

@Override
public long performDynamicTableTest(Configuration.AMOUNT_DATA amount_data) {
    long measuresData = 0;
    WebDriver webDriver = null;
    webDriver = new FirefoxDriver();
    webDriver.get("http://www.angularjs.org");
    switch (amount_data) {
        case 1DATA: dynamicTableList = webDriver.findElement(By.xpath("//*[@id='wrap-container-main-table']/a[1]")); break;
        case 1kDATA: dynamicTableList = webDriver.findElement(By.xpath("//*[@id='wrap-container-main-table']/a[2]")); break;
        case 3kDATA: dynamicTableList = webDriver.findElement(By.xpath("//*[@id='wrap-container-main-table']/a[3]")); break;
        case 5kDATA: dynamicTableList = webDriver.findElement(By.xpath("//*[@id='wrap-container-main-table']/a[4]")); break;
        case 10kDATA: dynamicTableList = webDriver.findElement(By.xpath("//*[@id='wrap-container-main-table']/a[5]")); break;
        case 20kDATA: dynamicTableList = webDriver.findElement(By.xpath("//*[@id='wrap-container-main-table']/a[6]")); break;
        case 50kDATA: dynamicTableList = webDriver.findElement(By.xpath("//*[@id='wrap-container-main-table']/a[7]")); break;
        case 1MDATA: dynamicTableList = webDriver.findElement(By.xpath("//*[@id='wrap-container-main-table']/a[8]")); break;
    }
    dynamicTableList.click();
}
do {
    React.performDynamicTableTest();
}

```

Rys.6 Prezentacja części projektu napisanej w selenium

The screenshot shows the IntelliJ IDEA interface with the following details:

- Project View:** Shows the project structure under "BackendJava".
- Code Editor:** Displays the `TestController.java` file, which contains several `@GetMapping` annotations for different endpoints like `/data0k`, `/data1m`, `/data5m`, `/data10m`, and `/data100m`.
- Run Tab:** Shows the application is running with port 10000.
- Console Tab:** Displays logs from the application's initialization, including messages about initializing Spring DispatcherServlet and completing initialization in 9 ms.
- Event Log:** Shows no events.

Rys.7 Prezentacja części back-end projektu przygotowaną w Spring Boot

```
DYNAMIC_TABLE_1k
Start: 1591025906839 Stop: 1591025906854 Diff: 15
Start: 1591025907470 Stop: 1591025907478 Diff: 8
Start: 1591025908105 Stop: 1591025908113 Diff: 8
Start: 1591025908738 Stop: 1591025908747 Diff: 9
Start: 1591025909372 Stop: 1591025909381 Diff: 9
DYNAMIC_TABLE_10k
Start: 1591025910007 Stop: 1591025910019 Diff: 12
Start: 1591025910655 Stop: 1591025910662 Diff: 7
Start: 1591025911287 Stop: 1591025911294 Diff: 7
Start: 1591025911920 Stop: 1591025911928 Diff: 8
Start: 1591025912556 Stop: 1591025912568 Diff: 12
DYNAMIC_TABLE_1m
Start: 1591025913205 Stop: 1591025913213 Diff: 8
Start: 1591025913839 Stop: 1591025913848 Diff: 9
Start: 1591025914471 Stop: 1591025914479 Diff: 8
Start: 1591025915105 Stop: 1591025915116 Diff: 11
Start: 1591025915734 Stop: 1591025915742 Diff: 8
ReactJS | Google Chrome | Tekst statyczny -> próbek: 5; średni czas: 1
ReactJS | Google Chrome | Tabela statyczna -> próbek: 5; średni czas: 484
ReactJS | Google Chrome | Tabela dynamiczna 1 dana -> próbek: 5; średni czas: 48
ReactJS | Google Chrome | Tabela dynamiczna 1k danych -> próbek: 5; średni czas: 9
ReactJS | Google Chrome | Tabela dynamiczna 10k danych -> próbek: 5; średni czas: 9
ReactJS | Google Chrome | Tabela dynamiczna 1m danych -> próbek: 5; średni czas: 8
```

Rys.8 Przykładowe wyniki uzyskiwane za pomocą Selenium

6. Uzyskane pomiary oraz wykresy.

Poniżej widzimy wyniki naszych pomiarów:

Wykonano celem dokładniejszych pomiarów po 5 próbek:

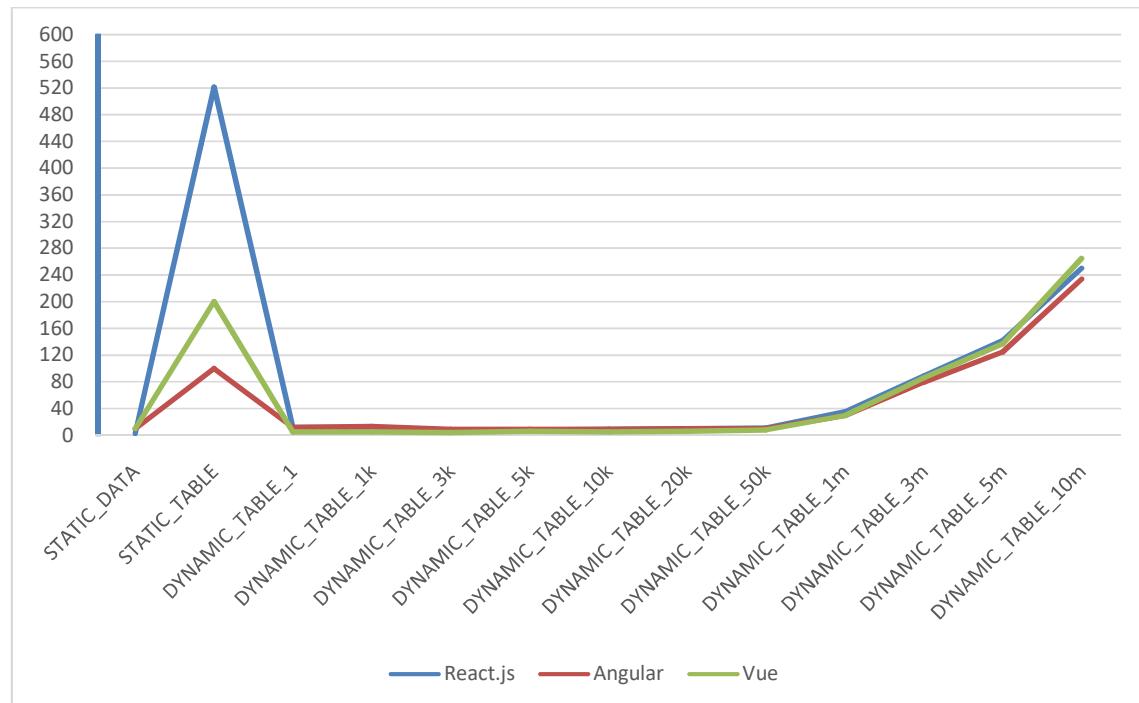
RODZAJE_POMIARÓW	React.Js	Angular	Vue.js
STATIC_DATA	5	12	20
STATIC_DATA	2	6	1
STATIC_DATA	2	16	9
STATIC_DATA	4	7	1
STATIC_DATA	3	10	1
STATIC_TABLE	557	2	197
STATIC_TABLE	684	1	154
STATIC_TABLE	450	1	220
STATIC_TABLE	451	1	200
STATIC_TABLE	466	1	200
DYNAMIC_TABLE	20	11	4
DYNAMIC_TABLE	11	10	8
DYNAMIC_TABLE	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

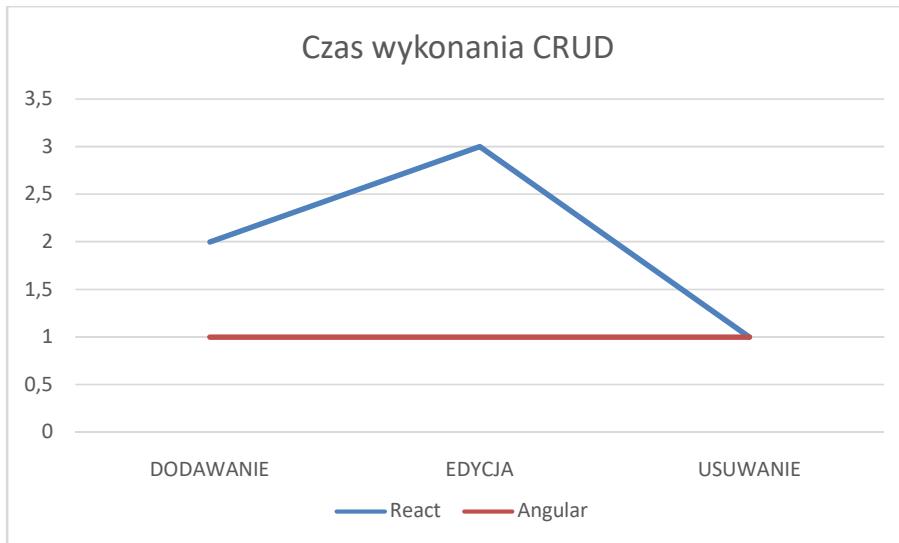
Przedstawienie danych na wykresie w zależności ms do wykonywanej operacji:



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.



7. Wnioski

Na podstawie uzyskanych danych oraz wiedzy sporządzmy wnioski na temat frameworków: Angular, Vue.js, React.js.

Jak widzimy wyniki przedstawiają przewagę Angular oraz Vue nad pozostałymi narzędziami. Angular to pełnowartościowy framework do tworzenia oprogramowania niewymagający 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 że osiągnął on wręcz świetne wyniki w porównaniu do konkurencji. Niestety mała popularność oraz walka z narzędziami wspieranymi przez wielkie korporacje sprawia że 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 pójdzie inną drogą w projekcie i wykorzystanie innych zależności pisania aplikacji (wykorzystanie sposobu pisania aplikacji z wykorzystaniem React Hooks lub Redux oraz Thunk) React.js pozwoliłoby na osiągnięcie 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 związań 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 nadążyć 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 społeczność React w pełni zrekompensowała reputację Google i upodobiła React do Angulara.

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