JURNAL RISET TEKNIK KOMPUTER
Halaman Jurnal: https://journal.smartpublisher.id/index.php/jurtikom
Halaman UTAMA Jurnal : https://journal.smartpublisher.id

DOI: https://doi.org/10.69714/f2h1kn88

PERBANDINGAN FRAMEWORK SVELTE JS DENGAN REACT JS TERHADAP PERFORMA WEB APPLICATION BERBASIS JAVASCRIPT

Zulhijaya [a*], Mustari S Lamada [b], Abdul Wahid [c]
[a] Fakultas Teknik, 32.zulhijaya@gmail.com, Universitas Negeri Makassar, Makassar, Sulawesi Selatan
[b] Fakultas Teknik, mustarilamada@gmail.com, Universitas Negeri Makassar, Makassar, Sulawesi Selatan
[c] Fakultas Teknik, wahid@unm.ac.id, Universitas Negeri Makassar, Makassar, Sulawesi Selatan
[*]Korespondensi

ABSTRACT

Comparison of Svelte JS Framework with React JS on Javascript-Based Web Application Performance. Informatics and Computer Engineering Education Study Program, Informatics and Computer Engineering Department, Faculty of Engineering, Makassar State University, (supervised by Mustari Lamada and Abdul Wahid). Comparison of Svelte JS and React JS frameworks using GTMetrix and WebPageTest tools by taking performance parameters including first contentful paint, speed index, large contentful paint, time to interactive, total blocking time, and cumulative layout shift. Each framework was tested by creating a 3-page website that had different loads. The test results were converted into scores where the score range was 1-3 for each parameter tested. React JS gets a total score of 26, 25, and 28 for each page. Svelte JS gets a total score of 34, 34, and 33 for each page. Native JS gets a total score of 34, 33, and 33 for each page. From these scores, it can be concluded that the Svelte JS framework is faster than React JS.

Keywords: *Framework Svelte JS, Framework React JS, Website Performance*

Abstrak

Perbandingan Framework Svelte JS dengan React JS Terhadap Performa Web Application Berbasis Javascript. Program Studi Pendidikan Teknik Informatika dan Komputer, Jurusan Teknik Informatika dan Komputer, Fakultas Teknik, Universitas Negeri Makassar, (dibimbing oleh Mustari Lamada dan Abdul Wahid). Perbandingan framework Svelte JS dan React JS menggunakan tools GTMetrix dan WebPageTest dengan mengambil parameter performance antara lain first contentful paint, speed index, large contentful paint, time to interactive, total blocking time, dan cummulative layout shift. Masing-masing framework diuji dengan membuat website 3 halaman yang memiliki load berbeda. Hasil pengujian dikonversi menjadi skor dimana rentang skor 1-3 dari setiap parameter yang diuji. React JS mendapatkan total skor masing-masing halaman antara lain 26, 25 dan 28. Svelte JS mendapatkan total skor masing-masing halaman antara lain 34, 34, dan 33. Native JS mendapatkan total skor masing-masing halaman antara lain 34, 33, dan 33. Dari skor tersebut disimpulkan bahwa framework Svelte JS lebih cepat dibandingkan dengan React JS.

Kata Kunci: *Framework Svelte JS, Framework React JS, Performa Website*

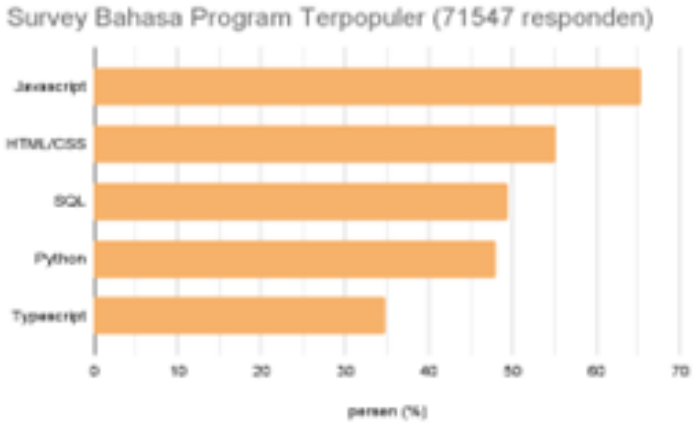
1. PENDAHULUAN

Website merupakan kumpulan halaman untuk menyimpan informasi layaknya sebuah buku dengan tujuan tertentu, informasi tersebut tersimpan pada sebuah domain di internet yang dapat diakses oleh semua orang melalui browser pada smartphone ataupun komputer dengan menuliskan *Uniform Resource Locator* (URL) website terkait.

Received April 25, 2025; Revised May 19, 2025; Accepted Juni 25, 2025; Published Juni 28, 2025

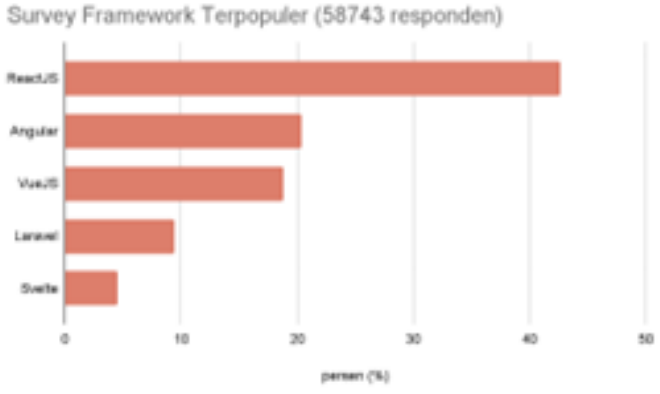
Seiring dengan perkembangan teknologi internet, website juga mengalami peningkatan. *Website* terbagi menjadi 3 generasi antara lain, generasi web 1.0 dimana pengunjung hanya dapat mencari dan melihat informasi pada halaman *website*. Generasi web 2.0 dapat melakukan komunikasi dua arah. Generasi web 3.0 mengalami peningkatan mulai dari segi penampilan hingga performa, generasi ini dapat menyesuaikan tampilan saat diakses melalui *desktop browser* dan *mobile browser*.

Perkembangan bahasa pemrograman *website* juga tak luput dari peningkatan, salah satunya JavaScript. Bahasa ini mulai mengalami peningkatan pesat saat salah satu perusahaan besar pada tahun 2013 yaitu Facebook memperkenalkan sebuah *framework* atau *library* JavaScript bernama React JS. Sejak React JS versi 0.3.0 tahun 2013 hingga React JS versi 16.8.6 tahun 2019, banyak *framework* JavaScript lain yang bermunculan diantaranya yang populer digunakan adalah Vue JS tahun 2014 dan Svelte JS tahun 2016.



Gambar 1. Hasil survei mengenai bahasa program yang banyak digunakan pada tahun 2022 (sumber: stackoverflow.com)

Dengan banyaknya jumlah *framework* JavaScript yang populer digunakan menyebabkan *programmer* ingin beralih untuk lebih efektif untuk penulisan sintaks. Ada 2 pilihan yang dimana saat ini *framework* ini popular di kalangan *programmer*. Pertama, React JS. React JS memiliki perbedaan yang cukup jauh dibandingkan dengan JavaScript dasar serta memiliki tingkat kesulitan yang tinggi untuk dipelajari. Kedua, Svelte JS. Untuk penulisan sintaksnya lebih sederhana. Sehingga Svelte JS merupakan bahasa yang mudah diadaptasi. Namun terlepas dari kemudahan Svelte JS tersebut, Svelte JS merupakan *framework* yang lebih muda/baru dibandingkan dengan React JS yang lebih dulu dirilis. React JS juga saat ini merupakan salah satu *framework* JavaScript yang populer.



Gambar 2. Hasil survei mengenai *framework* yang banyak digunakan pada tahun 2022 (sumber: stackoverflow.com)

Stackoverflow.com merupakan situs publik komunitas *developer* yang terpercaya. Situs ini terkenal dan sering digunakan oleh *programmer* profesional. Dari survei tahunan tahun 2022 menyatakan bahwa JavaScript merupakan bahasa pemrograman terpopuler tahun 2022 dan selama 10 tahun berturut-turut.

Sedangkan untuk *framework* javascript NodeJS menjadi terpopuler, ReactJS berada posisi kedua, dan SvelteJS berada pada posisi 16.

Saat ini, beberapa software *framework* menjadi lebih populer dibandingkan bahasa programnya sendiri. Bahkan di tingkat perusahaan, adopsi *framework* masih terus berkembang (Along, 2019). Membuat keputusan dalam menentukan framework yang digunakan dalam mengembangkan *software* masih menjadi permasalahan umum seorang *programmer*. Banyak faktor yang menjadi poin pertimbangan seperti dukungan komunitas, dokumentasi, performa, dan tingkat kesulitan. sehingga perlu dilakukan penelitian secara ilmiah mengenai perbandingan framework sehingga dapat menjadi referensi dalam mengetahui kekurangan dan kelebihan masing-masing framework.

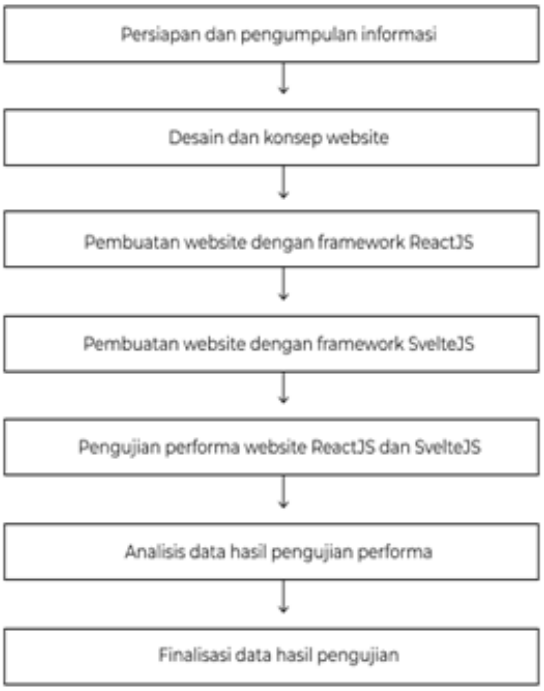
Dari beberapa penelitian yang relevan, telah ada yang membahas tentang perbandingan performa kinerja NodeJS, PHP, dan Phyton dalam aplikasi REST menyatakan bahwa NodeJS memiliki durasi respon yang paling cepat, sedangkan PHP adalah bahasa pemrograman yang memiliki performa umum paling baik (Rompis and Aji, 2018). Kemudian penelitian lain membahas tentang *Speed performance comparison of javascript MVC Frameworks* menyatakan bahwa dari semua test yang dilakukan, *framework* Angular 2.0 hampir selalu berada pada posisi teratas (Svensson, 2014). Kemudian ada penelitian lain tentang Studi perbandingan *website* dengan view *framework* (vueJS) dengan vanillaJS menyatakan bahwa *website* yang dibangun hanya menggunakan javascript biasa/vanillaJS memiliki performa yang lebih baik dan ukuran *bundle* yang lebih kecil daripada website yang dibangun menggunakan *framework* vueJS (Widianarko, 2018).

Dari penelitian yang ada sebelumnya, telah dilakukan perbandingan beberapa *framework* javascript antara lain Angular 1.5, Angular 2.0, Aurelia, Backbone.js, Ember, Knockout, Mithril, dan Vue. Namun, belum ada penelitian yang secara spesifik membahas tentang perbandingan performa antara ReactJS dan SvelteJS untuk *website*. Sementara itu, kedua *framework* tersebut merupakan *framework* yang banyak digunakan saat ini khususnya di kalangan profesional *web developer*. Sedangkan dari sisi akademis, pembelajaran pemrograman (*web programming*) merupakan salah satu ha1 yang patut mendapatkan perhatian (Mustari, 2015). Olehnya itu, dalam penelitian ini penulis ingin menguji performa dari *framework* SvelteJS dan ReactJS untuk kemudian dilakukan analisis perbandingan dari hasil performa kedua *framework* tersebut yang mana nantinya bisa menjadi referensi baik untuk rana profesional *web developer* maupun rana akademis.

2. METODOLOGI PENELITIAN

2.1. Metode Penelitian

Jenis penelitian yang digunakan oleh peneliti merupakan penelitian kuantitatif dengan menggunakan studi pendekatan komparatif. Penelitian ini untuk mengetahui perbandingan performa *web application* berbasis javascript yang menggunakan *framework* ReactJS dengan Svelte JS. penelitian ini akan yang terdiri dari beberapa tahap antara lain persiapan, pengumpulan data, validasi data, hingga analisis data. Kegiatan ini dilaksanakan secara langsung oleh peneliti tanpa melibatkan pihak lain.Adapun desain penelitian yang merupakan gambaran besar tahap-tahap penelitian yang akan dilaksanakan.



Gambar 3. Alur desain penelitian

1. Persiapan dan Pengumpulan Informasi
Tahap ini merupakan tahap awal dimana peneliti mengumpulkan informasi tentang *framework* SvelteJS dan ReactJS, seperti *best practices* dalam membuat *website* masing-masing *framework*, struktur penulisan *code*, cara kerja atau *lifecycle*, dan *component* yang bisa disediakan masing-masing *framework*.

2. Desain dan Konsep *Website*
Tahap ini peneliti merancang tampilan *website* yang sederhana menggunakan aplikasi desain Figma. Desain *website* dibutuhkan sebagai gambaran dalam membuat *website* pada tahap selanjutnya. halaman *website* hanya terdiri dari tiga skenario halaman yang mencakup/menampilkan bermacam-macam data yang akan dimuat dalam *website*. Tiga skenario yang dimaksud adalah sebagai berikut :

a. Skenario pertama, memuat halaman *website* pada menu beranda yang memuat konten teks beserta 1 konten gambar.
b. Skenario kedua, memuat halaman galeri yang menampung 10 gambar atau foto yang mana tiap gambarnya berukuran diatas 10 mb.
c. Skenario ketiga, memuat pemutar video dengan size video berukuran 50 mb.

3. Pembuatan *Website* dengan *Framework* SvelteJS
Tahap ini peneliti membuat *website* dengan menggunakan *text editor* Visual Studio Code dan Chrome *Web Browser*. Pembuatan *website* akan memanfaatkan dokumentasi dari situs resmi ReactJS dengan meminimalisir penggunaan *library* pihak ketiga sehingga hasil performa yang diuji pada tahap selanjutnya merupakan performa murni dari ReactJS itu sendiri. Pembuatan *website* dengan menggunakan ReactJS akan membutuhkan waktu sedikit lebih lama dibandingkan SvelteJS karena baris kode yang dituliskan dalam framework ReactJS lebih panjang.

4. Pembuatan *Website* dengan *Framework* ReactJS
Tools yang digunakan dalam pembuatan *website* SvelteJS sama dengan yang digunakan saat membuat ReactJS yaitu Visual Studio Code sebagai *text editor* dan Chrome *Web Browser* sebagai *browser*. Pembuatan *code* dengan SvelteJS membutuhkan waktu lebih sedikit dibandingkan dengan ReactJS. Namun, proses *compiling* dari SvelteJS akan butuh waktu lebih lama dibandingkan waktu *building* ReactJS, karena

SvelteJS harus merubah *code* yang ditulis dengan svelte menjadi *code* javascript terlebih dahulu sebelum bisa digunakan.

5. Pengujian Performa *Website* ReactJS dan SvelteJS

Tahap ini hasil dari *website* yang dibuat kemudian dilakukan pengujian performa menggunakan *tools* GTMetrix. Masing-masing dari kedua *tool* tersebut, peneliti mengambil 6 indikator performa, antara lain *first contentful paint, speed index, largest contentful paint, time to interactive, total blocking time, cumulative layout shift*.

Skenario pengujian penelitian ini, kedua *website* yang dibuat akan di *deploy* pada *environment server* yang sama di situs render.com. Peneliti juga akan membuat satu *website* dengan Javascript native sebagai *controller* atau pembanding. Kemudian dilakukan pengujian terhadap tiga *website* tersebut dengan *tools* GTMetrix untuk memperoleh data performa masing-masing *website* yang diuji.

6. Analisis data hasil pengujian performa

Tahap ini peneliti melakukan analisis dari hasil uji performa kedua *website*. Dari data pengujian dengan *tools* GTMetrix akan dibuat tabel hasil pengujian yang mencakup data dari tiga *website* yang diuji, beserta hasil data performanya.

7. Finalisasi data hasil pengujian

Tahap ini merupakan tahap akhir dari penelitian, peneliti melengkapi dokumentasi serta lampiran yang dibutuhkan dalam penulisan penelitian.

2.2. Metode Pengumpulan Data

Dalam penelitian ini terdapat 6 indikator yang diteliti dengan menggunakan instrumen penelitian yaitu GT Metrix. Data akan diuji dan dikumpulkan dari *website* yang telah dibuat oleh peneliti. *Website* terdiri dari tiga halaman berbeda yang memuat tipe data dan algoritma berbeda.

Tabel 1. Table indikator penelitian

No.	Indikator	Instrumen	Aspek diteliti
1	*First Contentful Paint*	GT Metrix	Waktu yang dibutuhkan *browser* untuk merender bagian pertama konten DOM.
2	*Speed Index*	GT Metrix	Seberapa cepat konten ditampilkan secara visual.
3	*Largest Contentful Paint*	GT Metrix	Waktu yang dibutuhkan untuk memuat konten terbesar.
4	*Time to Interactive*	GT Metrix	Waktu yang dibutuhkan hingga halaman sepenuhnya dapat berinteraksi dengan pengguna.
5	*Total Blocking Time*	GT Metrix	Jumlah waktu yang terblokir antara input pengguna hingga keseluruhan halaman dimuat.
6	*Cumulative Layout Shift*	GT Metrix	Seberapa sering elemen pada *website* terjadi pergeseran karena adanya elemen yang lambat untuk dimuat.

Skor setiap item indikator bisa mendapatkan total skor tertinggi adalah 3 poin dan terendah 1 poin. Dengan rincian, tiap instrumen dapat menghasilkan skor 1 - 3 poin. Jika semua indikator mendapatkan poin penuh maka maksimal skor yang bisa didapatkan dari 1 website sebanyak 54 poin.

2.3. Teknik Analisis Data

Teknik analisis data yang digunakan adalah teknik analisis deskriptif. Teknik analisis data merupakan cara yang digunakan untuk menguraikan keterangan-keterangan atau data yang diperoleh agar data dipahami bukan saja oleh orang yang mengumpulkan data (peneliti) tetapi juga oleh orang lain. Analisis data merupakan kegiatan setelah data dari terkumpul. Kemudian, data-data yang didapat dari hasil pengujian lalu diuraikan berdasarkan skenario pengujian yang dilakukan.

Peneliti menggunakan satuan skor 1 sampai 3 dengan ketentuan, saat status kecepatan dari instrumen adalah Cepat/Bagus mendapatkan skor 3, untuk status kecepatan dari instrumen adalah Menengah/Butuh Peningkatan mendapatkan skor 2, dan untuk status kecepatan dari instrumen adalah Lambat/Buruk mendapatkan skor 1.

3. HASIL DAN PEMBAHASAN

Data hasil performa website dengan sveltejs dan reactjs diperoleh dari parameter performance pada tools GTMetrix. Masing-masing website dikembangkan dengan design inteface yang sama dengan memuat tiga jenis halaman berbeda antara lain halaman home yang memuat content gambar dan teks, halaman gallery yang memuat content 14 gambar dengan ukuran 800x400 pixels, dan halaman video yang memuat content pemutar video. Tujuan dari load yang berbeda agar pengujian website lebih akurat dibanding hanya diuji pada satu load saja.

3. 1. Hasil performa *website* dengan react js

Berdasarkan penelitian yang dilakukan, peneliti mengumpulkan data performa website ReactJS dengan menggunakan tool GTMetrix. Pengujian menggunakan GTMetrix dilakukan sebanyak lima kali repetisi pada tiga jenis halaman berbeda, antara lain halaman *home*, halaman *gallery*, dan halaman video. Hasil pengujian dapat dilihat pada tabel dibawah.

Tabel 2. Data hasil pengujian performa *website* react js untuk halaman *home*

Indikator	Pengujian (detik)					Rata-rata
	1	2	3	4	5	
First Contentful Paint	2	2,10	2,70	1,9	3,5	2,44
Speed Index	2,50	2,20	2,80	2,1	3,8	2,68
Largest Contentful Paint	2,80	2,40	2,90	2,2	4,1	2,88
Time to Interactive	2,00	2,10	2,70	1,9	3,5	2,44
Time Blocking Time	0,00	0,00	0,00	0	0	0,00
Cummulative Layout Shift	0,21	0,21	0,21	0,21	0,21	0,21

Tabel 3. Skor hasil pengujian performa *website* react js untuk halaman *home*

Indikator	Rata-rata (detik)	Status	Skor
First Contentful Paint	2,44	Menengah	2
Speed Index	2,68	Cepat	3
Largest Contentful Paint	2,88	Menengah	2
Time to Interactive	2,44	Cepat	3
Time Blocking Time	0,00	Cepat	3

Cummulative Layout Shift	0,21	Butuh Peningkatan	2
		Total	**15**

Setelah mendapatkan rata-rata dari masing-masing indikator yang diuji, kemudian dilakukan konversi skor dari rata-rata pada tiap indikator. Indikator *first contentful paint* dengan rata-rata 2,44 detik mendapatkan hasil menengah dengan skor 2. Indikator *speed index* dengan rata-rata 2,68 detik mendapatkan hasil cepat dengan skor 3. Indikator *largest contentful paint* dengan rata-rata 2,88 detik mendapatkan hasil menengah dengan skor 2. Indikator *time to interactive* dengan rata-rata 2,44 detik mendapatkan hasil cepat dengan skor 3. Indikator *total blocking time* dengan rata-rata 0 detik mendapatkan hasil cepat dengan skor 3. Indikator *cummulative layout shift* dengan rata-rata 0,21 mendapatkan hasil butuh peningkatan dengan skor 2.

Tabel 4. Data hasil pengujian performa *website* react js untuk halaman *gallery*

Indikator	Pengujian (detik)					Rata-rata
	1	2	3	4	5	
First Contentful Paint	2,2	2,60	2,70	1,8	1,8	2,22
Speed Index	2,70	3,20	3,20	2,4	2,4	2,78
Largest Contentful Paint	2,40	3,40	3,00	2,1	2,4	2,66
Time to Interactive	2,20	2,60	2,70	1,8	1,8	2,22
Time Blocking Time	0,00	0,00	0,00	0	0	0,00
Cummulative Layout Shift	0,78	0,57	0,46	0,66	0,67	0,63

Tabel 5. Skor hasil pengujian performa *website* react js untuk halaman *gallery*

Indikator	Rata-rata (detik)	Status	Skor
First Contentful Paint	2,22	Menengah	2
Speed Index	2,78	Cepat	3
Largest Contentful Paint	2,66	Menengah	2
Time to Interactive	2,22	Cepat	3
Time Blocking Time	0,00	Cepat	3
Cummulative Layout Shift	0,63	Buruk	1
		Total	**14**

Tabel 6. Data hasil pengujian *website* react js untuk halaman video

Indikator	Pengujian (detik)					Rata-rata
	1	2	3	4	5	
First Contentful Paint	2,9	2,70	2,20	2,6	2,6	2,6
Speed Index	3,00	2,70	2,30	2,6	2,6	2,64
Largest Contentful Paint	2,90	2,70	2,20	2,6	2,6	2,60
Time to Interactive	2,90	2,70	2,20	2,6	2,6	2,60
Time Blocking Time	0,00	0,00	0,00	0	0	0,00
Cummulative Layout Shift	0,00	0,00	0,00	0	0	0,00

Tabel 7. Skor hasil pengujian website react js untuk halaman video

Indikator	Rata-rata (detik)	Status	Skor
First Contentful Paint	2,6	Menengah	2
Speed Index	2,64	Cepat	3
Largest Contentful Paint	2,60	Menengah	2
Time to Interactive	2,60	Cepat	3
Time Blocking Time	0,00	Cepat	3
Cummulative Layout Shift	0,00	Bagus	3
		Total	**16**

3.2. Hasil performa *website* dengan svelte js

Berdasarkan penelitian yang dilakukan, penulis mengumpulkan data performa website SvelteJS dengan menggunakan tool GTMetrix. Pengujian menggunakan GTMetrix dilakukan sebanyak lima kali repetisi pada tiga jenis halaman berbeda, antara lain halaman home, halaman gallery, dan halaman video. Hasil pengujian dapat dilihat pada tabel dibawah.

Tabel 8. Data hasil pengujian performa *website* untuk halaman *home*

Indikator	Pengujian (detik)					Rata-rata
	1	2	3	4	5	
First Contentful Paint	1	1,00	1,40	1,6	1	1,2
Speed Index	1,40	1,10	1,80	2,1	1,4	1,56
Largest Contentful Paint	1,70	1,20	2,10	2,4	0,7	1,82
Time to Interactive	1,00	1,20	1,60	1,8	1,2	1,36
Time Blocking Time	0,00	0,00	0,00	0	0	0,00
Cummulative Layout Shift	0,21	0,00	0,21	0,21	0	0,13

Tabel 9. Skor hasil pengujian performa *website* untuk halaman *home*

Indikator	Rata-rata (detik)	Status	Skor
First Contentful Paint	1,2	Cepat	3
Speed Index	1,56	Cepat	3
Largest Contentful Paint	1,82	Cepat	3
Time to Interactive	1,36	Cepat	3
Time Blocking Time	0,00	Cepat	3
Cummulative Layout Shift	0,13	Butuh Peningkatan	2
		Total	**17**

Tabel 10. Data hasil pengujian performa *website* untuk halaman *gallery*

Indikator	Pengujian (detik)					Rata-rata
	1	2	3	4	5	
First Contentful Paint	0,94	1,30	1,40	1,4	1,4	1,288
Speed Index	1,00	1,40	1,50	1,4	1,4	1,34
Largest Contentful Paint	1,20	1,30	1,60	1,4	1,4	1,38
Time to Interactive	1,10	1,30	1,40	1,5	1,6	1,38
Time Blocking Time	0,00	0,00	0,00	0	0	0,00
Cummulative Layout Shift	0,64	0,67	0,00	0,01	0,89	0,44

Tabel 11. Skor hasil pengujian performa *website* untuk halaman *gallery*

Indikator	Rata-rata (detik)	Status	Skor
First Contentful Paint	1,288	Cepat	3
Speed Index	1,34	Cepat	3
Largest Contentful Paint	1,38	Cepat	3
Time to Interactive	1,38	Cepat	3
Time Blocking Time	0,00	Cepat	3
Cummulative Layout Shift	0,44	Buruk	1
		Total	**16**

Tabel 12. Data hasil pengujian performa *website* untuk halaman video

Indikator	Pengujian (detik)					Rata-rata
	1	2	3	4	5	
First Contentful Paint	1,4	1,40	0,99	0,97	1	1,152
Speed Index	4,50	3,40	3,10	3,6	3,2	3,56
Largest Contentful Paint	1,40	1,40	0,99	0,97	1	1,15
Time to Interactive	1,50	1,60	1,10	1,1	1,2	1,30
Time Blocking Time	0,00	0,00	0,00	0	0	0,00
Cummulative Layout Shift	0,01	0,01	0,00	0,01	0,01	0,01

Tabel 13. Skor hasil pengujian performa *website* untuk halaman video

Indikator	Rata-rata (detik)	Status	Skor
First Contentful Paint	1,152	Cepat	3

Speed Index	3,56	Menengah	2
Largest Contentful Paint	1,15	Cepat	3
Time to Interactive	1,30	Cepat	3
Time Blocking Time	0,00	Cepat	3
Cummulative Layout Shift	0,01	Bagus	3
		Total	**17**

3.3. Hasil performa *website* dengan javascript native

Berdasarkan penelitian yang dilakukan, penulis mengumpulkan data performa website Javascript native dengan menggunakan tool GTMetrix. Pengujian menggunakan GTMetrix dilakukan sebanyak lima kali repetisi pada tiga jenis halaman berbeda, antara lain halaman home, halaman gallery, dan halaman video. Hasil pengujian dapat dilihat pada tabel dibawah.

Tabel 14. Data hasil pengujian performa *website* javascript native untuk halaman *home*

Indikator	Pengujian (detik)					Rata-rata
	1	2	3	4	5	
First Contentful Paint	1,1	0,98	0,97	0,99	1,4	1,088
Speed Index	1,60	1,60	1,10	1,4	1,5	1,44
Largest Contentful Paint	2,10	2,00	1,20	1,6	1,6	1,70
Time to Interactive	1,10	0,98	0,97	0,99	1,4	1,09
Time Blocking Time	0,00	0,00	0,00	0	0	0,00
Cummulative Layout Shift	0,22	0,22	0,00	0,22	0	0,13

Tabel 15. Skor hasil pengujian performa *website* javascript native untuk halaman *home*

Indikator	Rata-rata (detik)	Status	Skor
First Contentful Paint	1,088	Cepat	3
Speed Index	1,44	Cepat	3
Largest Contentful Paint	1,70	Cepat	3
Time to Interactive	1,09	Cepat	3
Time Blocking Time	0,00	Cepat	3
Cummulative Layout Shift	0,13	Butuh Peningkatan	2
		Total	**17**

Tabel 16. Data hasil pengujian performa *website* javascript native untuk halaman *gallery*

Indikator	Pengujian (detik)					Rata-rata
	1	2	3	4	5	
First Contentful Paint	1,2	1,20	1,30	1,7	1,2	1,32
Speed Index	1,60	1,50	1,60	1,9	1,5	1,62
Largest Contentful Paint	1,60	1,40	1,50	1,9	1,6	1,60
Time to Interactive	1,20	1,20	1,30	1,7	1,2	1,32
Time Blocking Time	0,00	0,00	0,00	0	0	0,00
Cummulative Layout Shift	0,24	0,12	0,94	0,34	0,86	0,50

Tabel 17. Skor hasil pengujian performa *website* javascript native untuk halaman *gallery*

Indikator	Rata-rata (detik)	Status	Skor
First Contentful Paint	1,32	Cepat	3
Speed Index	1,62	Cepat	3
Largest Contentful Paint	1,60	Cepat	3
Time to Interactive	1,32	Cepat	3
Time Blocking Time	0,00	Cepat	3
Cummulative Layout Shift	0,50	Buruk	1
		Total	**16**

Tabel 18. Data hasil pengujian performa *website* javascript native untuk halaman video

Indikator	Pengujian (detik)					Rata-rata
	1	2	3	4	5	
First Contentful Paint	1,7	1,70	1,20	1,6	1,6	1,56
Speed Index	4,40	3,60	4,30	4,8	3,8	4,18
Largest Contentful Paint	1,70	1,70	1,20	1,6	1,6	1,56
Time to Interactive	1,70	1,70	1,20	1,6	1,6	1,56
Time Blocking Time	0,00	0,00	0,00	0	0	0,00
Cummulative Layout Shift	0,01	0,01	0,01	0,01	0,01	0,01

Tabel 19. Skor hasil pengujian performa *website* javascript native untuk halaman video

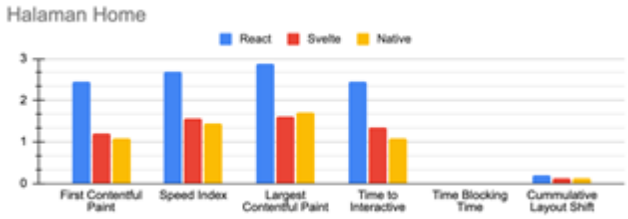
Indikator	Rata-rata (detik)	Status	Skor
First Contentful Paint	1,56	Cepat	3

Speed Index	4,18	Menengah	2
Largest Contentful Paint	1,56	Cepat	3
Time to Interactive	1,56	Cepat	3
Time Blocking Time	0,00	Cepat	3
Cummulative Layout Shift	0,01	Bagus	3
		Total	**17**

3.4. Perbandingan rata-rata indikator performa Svelte JS, React JS, dan JavaScript Native.
Rata-rata performa untuk masing-masing indikator pada halaman home dari website yang diuji dapat dilihat dari tabel di bawah. Lima dari enam indikator yang diuji, menunjukkan waktu yang dihasilkan *framework* React JS masih lebih tinggi dibandingkan dengan Svelte JS. Terutama pada indikator *first contentful paint* dan *largest contentful paint* yang memiliki selisih yang signifikan yaitu masing-masing selisih 1,24 detik dan 1,26 detik.

Tabel 20. Rata-rata tiap indikator dari ketika *framework* pada halaman *home*

Keterangan	React	Svelte	Native
First Contentful Paint	2,44	1,2	1,088
Speed Index	2,68	1,56	1,44
Largest Contentful Paint	2,88	1,82	1,70
Time to Interactive	2,44	1,36	1,09
Time Blocking Time	0,00	0,00	0,00
Cummulative Layout Shift	0,21	0,13	0,13



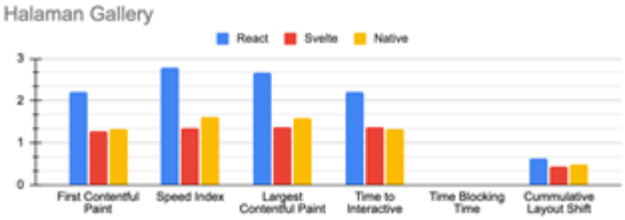
Gambar 3. Grafik rata-rata tiap indikator dari ketiga *framework* pada halaman *home*

Rata-rata performa untuk masing-masing indikator pada halaman gallery dari website yang diuji dapat dilihat dari tabel di bawah. Lima dari enam indikator yang diuji, menunjukkan waktu yang dihasilkan *framework* ReactJS masih lebih tinggi dibandingkan dengan SvelteJS. Terutama pada indikator *first contentful paint* dan *largest contentful paint* yang memiliki selisih yang signifikan yaitu masing-masing selisih 0,932 detik dan 1,28 detik.

Tabel 21. Rata-rata tiap indikator dari ketiga *framework* pada halaman *gallery*

Keterangan	React	Svelte	Native
First Contentful Paint	2,22	1,288	1,32
Speed Index	2,78	1,34	1,62
Largest Contentful Paint	2,66	1,38	1,60

Time to Interactive	2,22	1,38	1,32
Time Blocking Time	0,00	0,00	0,00
Cummulative Layout Shift	0,63	0,44	0,50

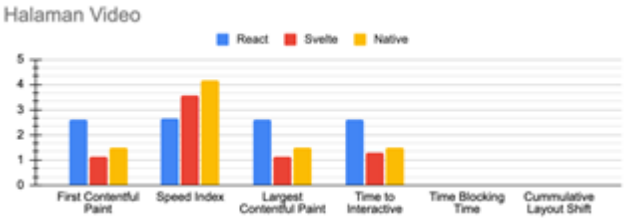


Gambar 4. Grafik rata-rata tiap indikator dari ketiga *framework* pada halaman *gallery*

Rata-rata performa untuk masing-masing indikator pada halaman video dari website yang diuji dapat dilihat dari tabel di bawah. Tiga dari enam indikator yang diuji, menunjukkan waktu yang dihasilkan *framework* ReactJS masih lebih tinggi dibandingkan dengan SvelteJS. ReactJS unggul pada indikator *Speed Index* dengan selisih 0,92 detik. Sedangkan SvelteJS masih tetap unggul signifikan pada *indikator first contentful paint* dan *largest contentful paint* dengan selisih masing-masing 1,448 detik dan 1,45 detik.

Tabel 22. Rata-rata tiap indikator dari ketiga *framework* pada halaman video

Keterangan	React	Svelte	Native
First Contentful Paint	2,6	1,152	1,56
Speed Index	2,64	3,56	4,18
Largest Contentful Paint	2,60	1,15	1,56
Time to Interactive	2,60	1,30	1,56
Time Blocking Time	0,00	0,00	0,00
Cummulative Layout Shift	0,00	0,01	0,01



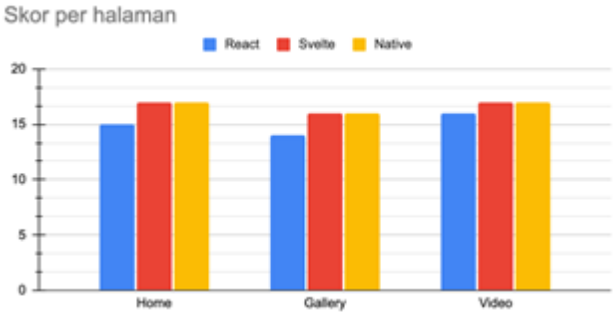
Gambar 5. Grafik rata-rata tiap indikator dari ketiga *framework* pada halaman video

3.5. Perbandingan Bobot Skor

Pembobotan skor berada pada rentang nilai 1 hingga 3 untuk masing-masing indikator yang diuji. Skor dari tiap indikator kemudian dibagi berdasarkan ketiga halaman yang diuji. Sehingga setiap halaman memiliki total skor masing-masing dari ketiga framework yang diuji.

Tabel 23. Total skor untuk tiap halaman

Halaman	React	Svelte	Native
Home	15	17	17
Gallery	14	16	16
Video	16	17	17



Gambar 5. Grafik total skor pada tiap halaman

3.5. Pembahasan

Kedua I javascript yang menjadi objek penelitian berjalan pada bahasa program dan *engine* yang sama, namun memiliki cara kerja yang berbeda. Selain itu, SvelteJS dan ReactJS juga berbeda dari cara penulisan kode perintah, yang mana SvelteJS memiliki kode yang lebih ramping dibandingkan ReactJS. Peneliti membuat tiga *website* berbeda dengan konten yang sama terhadap kedua framework (SvelteJS dan ReactJS) ditambah dengan Javascript Native (*non-framework*) sebagai pembanding. Konten *website* terdiri dari tiga jenis konten, antara lain konten text dan gambar, konten *gallery*, dan konten video.

Halaman dengan konten text dan gambar tercatat lebih cepat pada *website* SvelteJS saat diuji. Lima dari enam indikator yang diuji dari *website* SvelteJS memiliki skor lebih tinggi dari indikator pengujian website ReactJS. Lima indikator tersebut antara lain *first contentful paint* dengan waktu 1,2 detik, *speed index* dengan waktu 1,56 detik, *largest contentful paint* dengan waktu 1,62 detik, *time to interactive* dengan waktu 1,36 detik dan *cumulative layout shift* dengan skor 0,13. Sedangkan halaman konten text dan gambar pada *website* Javascript Native tercatat sedikit lebih tinggi dibandingkan dengan *website* SvelteJS. Website Javascript Native lebih cepat pada tiga dari enam indikator, antara lain *first contentful paint* dengan waktu 1,088 detik, *speed index* dengan waktu 1,44 detik dan *time to interactive* dengan waktu 1,09 detik.

Halaman dengan konten *gallery* pada *website* SvelteJS tercatat lebih cepat dibandingkan dengan *website* ReactJS saat dilakukan pengujian, lima dari enam indikator SvelteJS memiliki kecepatan lebih tinggi, antara lain indikator *first contentful paint* dengan waktu 1,288 detik, *speed index* dengan waktu 1,34 detik, *largest contentful paint* dengan waktu 1,38 detik, *time to interactive* dengan waktu 1,38 detik dan *cummulative layout shift* dengan skor 0,44. Sedangkan untuk *website* Javascript Native, *website* SvelteJS masih lebih unggul dibandingkan dengan *website* Javascript Native, empat dari enam indikator pengujian tercatat memiliki kecepatan lebih tinggi, antara lain indikator *first contentful paint* dengan waktu 1,288 detik, *speed index* dengan waktu 1,34 detik, *largest contentful paint* dengan waktu 1,38 detik dan *cumulative layout shift* dengan skor 0,44.

Halaman dengan konten video pada website SvelteJS tercatat memiliki kecepatan lebih tinggi dibandingkan ReactJS. SvelteJS unggul tiga dari enam indikator pengujian, antara lain indikator *first contentful paint* dengan waktu 1,152 detik, *largest contentful paint* dengan waktu 1,15 detik dan *time to interactive* dengan waktu 1,3 detik. *Website* ReactJS lebih cepat pada indikator *speed index* dibandingkan dengan *website* SvelteJS dan Javascript Native dengan waktu 2,64 detik. Sedangkan untuk SvelteJS lebih unggul dibandingkan Javascript Native. Saat dilakukan pengujian, empat dari enam indikator *website* SvelteJS tercatat memiliki kecepatan lebih tinggi, antara lain indikator *first contentful paint* dengan waktu 1,152 detik, *speed index* dengan waktu 3,56 detik, *largest contentful paint* dengan waktu 1,15 detik, dan *time to interactive* dengan waktu 1,3 detik.

4. KESIMPULAN DAN SARAN

Berdasarkan hasil penelitian dan pembahasan, terdapat kesimpulan dan saran dari peneliti, antara lain:

a. *Website* yang dibangun memiliki tiga jenis halaman antara lain halaman *home*, *gallery*, dan video. Untuk hasil skoring dari *website* yang dibangun dengan *framework* SvelteJS memiliki skor untuk halaman *home* yaitu 17, skor untuk halaman *gallery* yaitu 16, dan skor untuk halaman video yaitu 17.
b. *Website* yang dibangun menggunakan *framework* ReactJS memiliki jenis dan jumlah halaman yang sama persis dengan SvelteJS. Untuk hasil skoring dari *website* yang dibangun dengan framework

ReactJS memiliki skor untuk halaman *home* yaitu 15, skor untuk halaman *gallery* yaitu 14, dan skor untuk halaman video yaitu 16.

c. Peneliti juga membuat *website* dengan menggunakan javascript native sebagai controller dan pembanding tambahan. *website* yang dibuat memiliki jenis dan jumlah halaman yang sama dengan kedua *framework* di atas. Untuk hasil skoring dari *website* yang dibangun dengan javascript native memiliki skor untuk halaman *home* yaitu 17, skor untuk halaman *gallery* yaitu 16, dan skor untuk halaman video yaitu 17. Dari hasil di atas maka kesimpulan dari penelitian ini adalah secara keseluruhan *website* yang berjalan dengan framework SvelteJS memiliki *load* lebih cepat dibandingkan dengan ReactJS. Sedangkan *website* dengan Javascript native dan SvelteJS memiliki *load* yang sama cepat.

Saran

Saran untuk peneliti yang ingin mengembangkan penelitian ini adalah membuat lebih banyak model *website* yang ingin diuji, misalnya menambahkan halaman *game rendering* atau menambahkan halaman yang memuat banyak animasi secara bersamaan. Selain itu juga peneliti yang ingin mengembangkan penelitian ini dapat membandingkan *framework* yang diuji disini dengan *framework* javascript lain yang lebih baru.

Saran untuk *developer*, dalam memilih *framework* yang digunakan dalam sebuah project, bukan hanya performa yang perlu diperhatikan meskipun performa juga merupakan unsur penting. Namun yang perlu diperhatikan selain performa adalah komunitas dari *framework* tersebut, *maintainer* dari *framework* tersebut siapa dan apakah masih aktif, dan umur dari *framework* tersebut.

DAFTAR PUSTAKA

[1] Abbas, Wahidin. (2013). *Analisa Kepuasan Mahasiswa Terhadap Website Universitas Negeri Yogyakarta (UNY).* Universitas Negeri Yogyakarta.
[2] Ankush, S.D. (2014). *XSS Attack Prevention Using DOM based filtering API.* National Institute of Technology Rourkela.
[3] Along, Daniel. (2019). *Framework is a must for better programming.*
[4] Andriyany, Dwi Peny. (2021). *Analisis Konsep Produktivitas Dan Faktor-Faktor Yang Mempengaruhi Produktivitas Kerja Karyawan (Studi Literatur).* STIE PGRI Dewantara.
[5] Chrome Developers. (2023). *Lighthouse.* https://developer.chrome.com/docs/lighthouse/
[6] Dicki, Ilhamsyah, Dian Prawira. (2020). *Implementasi Framework Accelerated Mobile Pages Pada Pengembangan Website Program Studi Sistem Informasi. Jurnal Komputer dan Aplikasi*, 8 (2), 67-78.
[7] GTMetrix. (2023). *How fast does your website load? Find out with GTmetrix.* GTMetrix. https://gtmetrix.com
[8] ISO/IEC 25010. (2012). *System and Software Quality Requirements and Evaluation (SQuaRE) – System and Software Quality Models', Canadian Standards Association.*
[9] Laurensius, dkk. (2017).*Platform E-Learning untuk Pembelajaran Pemrograman Web Menggunakan Konsep Progressive Web Apps.* Jurnal Teknik ITS. Vol 6. No 2. https://ejurnal.its.ac.id/index.php/teknik/article/viewFile/24291/4821
[10] Levlin, M. (2020). *DOM benchmark comparison of the front-end JavaScript frameworks React, Angular, Vue, and Svelte.* Abo Akamedi University.
[11] Margono. (2005) Metodologi Penelitian Pendidikan. Jakarta: PT Rineka Cipta.
[12] Mustari. (2015). *Pengembangan Model Project Based Learning Mata Kuliah Web Programming Pada Program Studi Pendidikan Teknik Informatika dan Komputer.* Universitas Negeri Makassar.
[13] Nursaid, dkk. (2020). *Pengembangan Sistem Informasi Pengelolaan Persediaan Barang Dengan ReactJS dan React Native Menggunakan Prototype (Studi Kasus: Toko Uda Fajri).* J-Ptik.Ub.ac.id.
[14] PageSpeed Insights. (2023). *PageSpeed Insights*. https://pagespeed.web.dev
[15] Sari, A.O., Abdilah, A., Sunarti. (2019). *Web Programming.* Yogyakarta : Graha Ilmu.
[16] Sianipar, S.H. (2015). *Pemrograman JavaScript: Teori dan Implementasi.*
[17] Svensson, Alexander. (2015). *Speed Performance Comparison of Javascript MVC Frameworks.* Faculty of Computing Blekinge Institute of Technology Karlskrona Sweden.
[18] Suanto, Leo. (2013). *Kiat Jitu Menulis Skripsi, Tesis dan Disertasi*. Jakarta : Erlangga.
[19] Sugiyono. (2010). *Metode Penelitian Pendidikan (Pendekatan kuantitatif, Kualitatif, dan R&D).* Bandung : Alfabeta.
[20] Sugiyono. (2014). *Statistik Untuk Penelitian*. Bandung : Alfabeta.

[21] Suharsimi, Arikunto. (2013). *Manajemen Penelitian*. Jakarta. Jakarta : Rineka Cipta.
[22] Rompis, A.C. dan Aji, R.F. (2018) 'Perbandingan Performa Kinerja Node.js, PHP, dan Python dalam Aplikasi REST', 4, p. 17.
[23] Rosady, Ruslan. (2010). *Metodologi Penelitian*. Jakarta : Rajawali Pers.
[24] Wali dan Ahmad. (2018). *Perancangan Access Open Journal System (AOJS) dengan menggunakan Framework Codeigniter dan ReactJs"*. Jurnal JTIK (Jurnal Teknologi Informasi dan Komunikasi).
[25] Wappalizer. (2023). *Identify Technologies on Websites*. Wappalyzer. https://wappalyzer.com
[26] Widianarko, Cahyo Wibowo. (2018). *Studi Perbandingan Website View Framework (Vue JS) dengan Vanilla JS.* Skripsi thesis, STMIK AKAKOM Yogyakarta.