Penerapan React JS Pada Pengembangan *FrontEnd* Aplikasi *Startup* Ubaform

Nasution
Jurusan Informatika Fakultas Teknologi Industri
Universitas Islam Indonesia
Yogyakarta
nasution@students.uii.ac.id

Lizda Iswari
Jurusan Informatika Fakultas Teknologi Industri
Universitas Islam Indonesia
Yogyakarta
lizda.iswari@uii.ac.id

***Abstract*—Perkembangan web untuk menunjang kebutuhan pada era pandemi COVID-19 sudah sangat banyak dijumpai. Banyak *startup* baru yang muncul dengan menawarkan keunggulan masing-masing di berbagai bidang seperti pendidikan, kesehatan, pekerjaan dan lain sebagainya. Hal ini mengharuskan perintis *startup* mampu bersaing merebut pasar pada bidang yang menjadi tujuan. Meski pelayanan dan kekayaan fitur menjadi hal utama dalam persaingan antara *startup*, namun performa sistem pada *startup* juga menjadi hal yang perlu dipertimbangkan. Masalah performa sebuah website yang lambat akan membuat pengunjung merasa tidak nyaman dan fatalnya akan menghilangkan keinginan pengunjung untuk kembali lagi mengakses layanan. Saat ini kebanyakan pembuatan sistem pada *startup* terutama pengembangan website dibangun dengan *Multi Page Application* (MPA) yang mana untuk performa nya lebih rendah dari pada penggunaan *Single Page Application* (SPA) yang saat ini dipercaya lebih cepat. Penerapan SPA akan meningkatkan kinerja serta pengurangan waktu dan biaya infrastruktur pengembangan. Pengembangan *front-end startup* ubaform berbasis SPA dengan menggunakan *library* javascript yaitu React JS. Penerapan React JS dalam pengembangan *startup* ubaform diharapkan memberikan kecepatan *rendering* yang tinggi sehingga performa pengaksesan data menjadi lebih cepat selain itu memberikan kemudahan kepada pengembang dalam membuat tampilan dengan bantuan *package*.**

Keywords— Javascript Library; Single Page Application; React JS

I. PENDAHULUAN

Dalam pengembangan sebuah teknologi tentunya banyak hal yang harus dipersiapkan sehingga teknologi yang dikembangkan dapat digunakan sebagaimana seharusnya. Pengembangan yang terencana secara sistematik akan meminimalisir kesalahan-kesalahan yang mungkin akan terjadi saat proses pengembangan. Dalam pengembangan *startup* ubaform perancangan terencana mengenai teknologi yang akan digunakan telah dipertimbangkan secara matang. Selain untuk mengurangi kesalahan, merancang pengembangan yang matang dapat membawa dampak pada kemudahan pengembangan *startup*.

Banyak teknologi yang telah ada yang dapat digunakan dalam pengembangan sebuah *startup* berbasis website. Penerapan teknologi-teknologi ini disesuaikan dengan kebutuhan dan tujuan dari website yang dibangun. misalnya ketika membangun sebuah website untuk kebutuhan menulis artikel, teknologi yang dibutuhkan tidak sekompleks dibandingkan ketika membangun website untuk penjualan *(e-commerce)* yang membutuhkan teknologi yang cepat dan mumpuni. hal ini perlu dipertimbangkan dan direncanakan dengan sebaik mungkin untuk kenyamanan pengguna dan untuk kemudahan serta kecepatan dalam pengembangan *startup* website. Selain itu teknologi yang digunakan haruslah mampu relevan untuk beberapa tahun kedepan. Hal ini akan berdampak pada biaya yang nantinya akan dikeluarkan oleh perusahaan, biasa nya untuk mempelajari dan menerapkan teknologi yang lebih baru membutuhkan pengeluaran yang lebih besar.

Berdasarkan Hal tersebut dalam perintisan *startup* ubaform saat ini menggunakan SPA. Selain lebih baru dari penggunaan MPA dan dengan performa yang lebih baik, Penggunaan SPA telah memiliki banyak dukungan pengembangan seperti *library* pendukung, komunitas, dokumentasi kode dan juga banyak perusahaan besar yang sudah beralih menggunakan SPA sehingga akan ada banyak pengembang yang memiliki *skill* pada bidang tersebut. SPA memuat sebagian besar sumber daya aplikasi (HTML/CSS/Javascript) hanya sekali. SPA tidak perlu memuat ulang seluruh halaman tetapi hanya sebagian data tertentu yang dibutuhkan pengguna. Oleh karena itu, hal ini mempercepat waktu yang dibutuhkan untuk menerima informasi tentang data yang diakses dan bahkan jumlah waktu yang diperlukan untuk mengakses fitur tertentu. Menurut survei, 20% pengguna akan meninggalkan situs web jika waktu pemuatan nya melebihi 3 detik. Selain itu, sebuah studi oleh Walmart telah mengungkapkan bahwa setiap penurunan 100 ms dalam waktu pemuatan menghasilkan pertumbuhan 1% dalam pendapatan tambahan [1].

MPA lebih lambat karena browser harus memuat ulang seluruh halaman dari awal setiap kali pengguna ingin mengakses data baru atau pindah ke bagian lain dari situs web. Waktu pemuatan optimal untuk situs web adalah 0,4 detik. Alasan lain penggunaan SPA pada web ubaform adalah karena banyaknya *resource* berupa gambar yang membutuhkan waktu pemuatan lebih lama jika menerapkan MPA yang harus memuat seluruh halaman web [2].

Banyak CTO *(Chief Technology Officer)* dan pengembang lainnya berdebat tentang *library* atau *framework* JavaScript mana yang harus dipilih untuk pengembangan website. Dari banyak nya *library* atau *framework* yang paling populer adalah Angular JS dan React JS. Masing-masing dari mereka memiliki kelebihan dan kelemahannya sendiri. Akan tetapi pada pengembangan *startup* ubaform penggunaan React JS lebih cocok dibandingkan menggunakan Angular JS. Hal ini dikarenakan keduanya memiliki perbandingan yang cukup signifikan, React JS merupakan *library* sedangkan Angular JS adalah *framework* [3]. Penggunaan *library* lebih fleksibel dibandingkan dengan *framework* yang sudah ditetapkan. Selain itu React JS berbasis komponen yang bisa digunakan kembali jika dibutuhkan hal ini dapat menghemat waktu pengembangan *startup* yang membutuhkan kecepatan pengembangan sehingga peluang menguasai pasar lebih dapat dijangkau lebih cepat.

II. KAJIAN PUSTAKA

Ubaform merupakan *startup* yang menyediakan layanan pengelolaan dokumen formulir, kuis, dan survei secara *online*. Dengan mengutamakan keunggulan *user interfaces* yang modern dan interaktif serta *rendering* yang cepat pada proses pembuatan serta penyajian ketiga fitur tersebut.

Oleh karena itu pada karya ilmiah ini menggunakan beberapa penelitian yang digunakan sebagai landasan penulisan atas kasus yang serupa. Meski dalam pemaparan tidak dijelaskan semua komponen yang dikembangkan karena batas halaman yang sedikit, Namun dapat mewakili pengembangan ubaform berbasis React JS.

2.1 *Single Page Application* (SPA)

Single Page Application (SPA) merupakan aplikasi yang berjalan di *browser* dan tidak memerlukan pemuatan ulang halaman saat digunakan. Ini berarti bahwa pengguna tidak berpindah halaman saat *user* mengirim setiap permintaan ke *server*. Menerapkan *single page application* memiliki dua manfaat utama yaitu mengurangi penggunaan *bandwidth* jaringan dan mempercepat penjelajahan. SPA ini didukung secara luas oleh pustaka Javascript yang mengurangi *bandwidth* jaringan. Data yang diterima dari *server* dalam format JSON *(JavaScript Object Notation)* dan dapat dilihat secara asinkron menggunakan JavaScript, sehingga penjelajahan menjadi lebih cepat [4].

2.2 React JS

React adalah *open-source library* JavaScript deklaratif, efisien dan fleksibel untuk membangun antarmuka pengguna. React memungkinkan untuk membuat *user interface* yang kompleks dengan set kode kecil yang terisolasi yang disebut "komponen". React JS ini digunakan untuk menangani lapisan tampilan dalam aplikasi satu halaman dan pengembangan *mobile application*. React JS dikelola oleh facebook, instagram, komunitas pengembang dan korporasi. React berusaha untuk memberikan kecepatan, kesederhanaan, dan skalabilitas. Beberapa fitur yang paling mencolok adalah JSX, Komponen *Stateful*, Model Objek Dokumen Virtual.

Seperti yang telah disinggung sebelumnya penggunaan React JS dalam pengembangan membuat proses pembuatan *user interface* interaktif menjadi lebih mudah. Penulisan *syntax* React JS yang deklaratif membuat alur kode menjadi lebih mudah terprediksi dan mudah untuk dilakukan *debug*. React JS berbasis komponen yang mana dapat membuat beberapa komponen yang terenkapsulasi sehingga mengatur *state*-nya sendiri, kemudian komponen tersebut digabungkan untuk membentuk *user interfaces* yang lebih kompleks. Hal ini juga memudahkan pengembang ketika melakukan *maintenance* ketika terjadi hal yang tidak sesuai karena pengembang akan langsung menuju komponen yang bermasalah tanpa mengganggu komponen lainya sehingga selain mudah juga lebih cepat. Selain itu komponen-komponen yang telah ditulis dapat digunakan kembali ketika pengembang membutuhkan hal ini akan menghemat waktu dan efisiensi kode agar tidak ditulis secara berulang-ulang.

2.3 JSX

JSX adalah sintaks seperti XML atau HTML yang digunakan oleh React untuk memperluas ECMAScript sehingga teks seperti XML atau HTML dapat digunakan berdampingan dengan kode JavaScript atau React. React memiliki logika *rendering* secara inheren digabungkan dengan logika UI lainnya, seperti bagaimana *event* ditangani, bagaimana status dapat berubah dari waktu ke waktu, dan bagaimana data dapat disiapkan untuk ditampilkan. JSX direkomendasikan untuk digunakan bersama dengan react dengan tujuan untuk menjelaskan bagaimana tampilan UI sehingga memungkinkan react untuk menampilkan pesan dan peringatan yang lebih berguna bagi pengembang. JSX memudahkan pengembang karena kode antara *markup* (HTML) dapat digabungkan langsung dengan logika (Javascript) sehingga penulisan sintaksnya lebih sederhana dan mudah daripada penulisan keduanya secara terpisah. Contoh penulisan JSX pada gambar 1:

```
const name = 'Budi';
const element = <h1>Halo, {name}</h1>;

ReactDOM.render(
  element,
  document.getElementById('root')
);
```

Gambar 1. Syntax JSX

2.4 *Node package manager* (NPM)

NPM merupakan *Node Package Manager* yang populer digunakan secara luas oleh pengembang JavaScript untuk berbagi tool, menginstal modul, dan mengelola dependensi. NPM dapat dilihat sebagai repositori untuk proyek *open source* bagi siapa saja yang ingin menggunakan dan mempublikasikan kode-kode tertentu. Hal Yang dapat dilakukan NPM adalah dapat menginstal dan meng-*uninstal package*, serta mengelola versi dan dependensi yang diperlukan untuk menjalankan proyek. NPM ada bersamaan dengan saat instalasi Node JS. Penggunaan NPM dalam proyek ubaform akan membantu dalam menginstal beberapa *package* yang dibutuhkan dalam pengembangan sehingga mempermudah pengembang serta menghemat waktu pengembangan *startup*.

III. METODOLOGI

Pada tahap ini dilakukan metodologi penerapan React JS antara lain tahapan perancangan dan implementasi Detail tahapan sebagai berikut:

A. Analisis dan Perancangan *front end* ubaform berbasis React JS

Tahap perancangan merupakan tahap yang sangat penting dalam pengembangan perangkat lunak dan menjadi dasar dalam proses pengembangan yang dilakukan. Dalam perancangan tampilan ubaform selain menggunakan React JS juga menggunakan beberapa *package* yang mendukung pengembangan serta kompleksitas dari tampilan ubaform yang dikembangkan. Penggunaan *package* dalam *library* React JS sudah sangat umum dilakukan oleh pengembang.

B. Implementasi React JS pada *front end* ubaform

Tahap selanjutnya adalah implementasi, tahap ini menjelaskan bagaimana implementasi dari React JS dan beberapa *package* bawaan yang digunakan dalam pengembangan. Selain itu juga menjelaskan bagaimana menggabungkan beberapa komponen *state* sehingga

membentuk tampilan yang kompleks serta menjelaskan penggunaan react router yang digunakan untuk *routing* ke beberapa halaman aplikasi.

IV. HASIL DAN PEMBAHASAN

Bab ini menjelaskan setiap langkah yang dijelaskan pada bab sebelumnya dengan lebih jelas.

A. Perancangan *front end* ubaform berbasis React JS

Seperti yang dijelaskan dalam metodologi, tahap ini akan menjelaskan hasil dan pembahasan dari perancangan sistem ubaform menggunakan React JS dan beberapa *package* dalam pengembangan *startup* ubaform. Dalam *paper* ini tidak akan dibahas semua rancangan tampilan yang akan dikembangkan mengingat batasan halaman yang dibatasi. Berikut tahapan analisis dan perancangan *front end* ubaform:

a. Metode Pengembangan

Pada pengembangan *startup* ubaform metode yang digunakan adalah scrum. Scrum merupakan salah satu teknik rekayasa perangkat lunak yang menggunakan prinsip agile, mengandalkan kekuatan kolaborasi tim, produk langkah demi langkah, dan proses berulang untuk mencapai hasil akhir.

b. Alur Kerja Aplikasi

Perancangan alur kerja aplikasi dilakukan untuk mengetahui bagaimana proses *user* menggunakan aplikasi. Hal ini akan membantu UI/UX memahami pembuatan tampilan serta tata letak fitur-fitur yang akan digunakan.

c. Desain Aplikasi

Sebelum tahap implementasi, dilakukan desain aplikasi dengan menggunakan aplikasi figma. Desain aplikasi berdasarkan perancangan alur kerja aplikasi sebelumnya. Perancangan desain aplikasi ubaform dilakukan agar nantinya mudah bagi pengembang dalam melakukan implementasi.

d. Teknologi pengembangan

Seperti yang telah dibahas sebelumnya teknologi untuk pengembangan ubaform menggunakan React JS dengan beberapa *package* tambahan. Penggunaan *package* dapat mempercepat proses pengembangan pembuatan *frontend* ubaform dikarenakan proses koding tidak dilakukan dari nol. Jadi seperti menggabungkan beberapa kode yang sudah ada dari *package* tertentu sesuai dengan kebutuhan. Dampak penggunaan *package* ini sangat signifikan dalam mempercepat proses pengembangan, bayangkan saja ketika pengembang ingin membuat komponen *header* atau membuat *sidebar* yang *responsive* dari nol menggunakan HTML dan CSS akan memakan waktu yang lama. Namun dengan bantuan *package* hal ini bisa teratasi. Berikut beberapa *package* tambahan yang digunakan:

- *Package* `react-router-dom` digunakan untuk dapat berpindah halaman dari halaman satu ke halaman lainya. Perpindahan halaman di sini tentu saja masih menganut *Single page application* yang mana perpindahan tampilan terjadi pada satu halaman tanpa melakukan *loading* atau pemuatan ulang halaman.
- *Package* `material UI`. Penggunaan Material UI membantu dalam mempercepat proses pembuatan *user interface*. material UI memiliki beberapa *package* berbeda antara lain `@material-ui/core`, *package* Material UI core tersedia berbagai komponen seperti *Button, Card, Form* dan komponen lainya. @material-ui/icons khusus menyediakan *icon* dan `@material-ui/lab` menyediakan komponen yang tidak ada pada `@material-ui/core`.
- *Package* `zxcvbn` yang berfungsi untuk membantu melakukan pengecekan terhadap kekuatan kombinasi sandi yang dimasukkan oleh *user*.
- *Package* `react-multi-carousel` yang berfungsi untuk membuat tampilan *card* bergerak (*carousel card*).
- *Package* `syncfusion/ej2-react-navigations` berfungsi dalam membantu pengembangan *dashboard* terutama pada navigasi *sidebar* untuk *user interface* ubaform.
- *Package* `react-icons` yang berfungsi menyediakan *icon* yang digunakan dalam pembuatan *user interfaces* jika *icon* pada `@material-ui/icons` kurang atau tidak tersedia.

B. Implementasi React JS pada front end ubaform

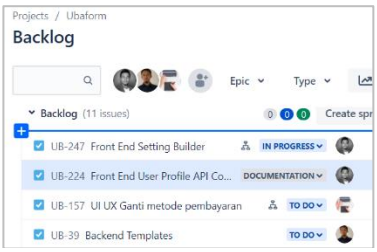
a. Metode pengembangan

Pada tahap ini dilakukan implementasi dari metode pengembangan scrum. Penggunaan metode *scrum* pada pengembangan *startup* ubaform diharapkan dapat meningkatkan kualitas dalam menghasilkan produk yang terbaik. Sebagai perintis *startup* yang memiliki banyak saingan kualitas produk haruslah bisa mengungguli kompetitor. Siklus pengembangan *scrum* dapat dilihat pada gambar 2 berikut:

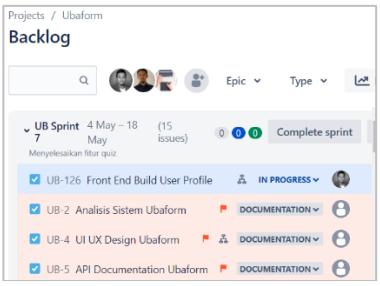
Gambar 2. proses scrum

Pada gambar 2 ada beberapa proses yang terdapat pada metode scrum. Proses dimulai dengan *vision* yang mana pada tahap ini akan dijabarkan tujuan pengembangan aplikasi. *User stories* untuk menggambarkan fitur yang ada pada aplikasi. *Sprint backlog* digunakan untuk menampilkan *list* pekerjaan yang akan dilakukan secara detail. *Daily scrum (standup daily)* digunakan untuk saling mengutarakan tentang apa yang sudah

dilakukan, akan dilakukan dan sedang dilakukan dalam 24 jam. *Sprint* digunakan untuk mengerjakan apa yang telah terdaftar pada *sprint backlog*. *Sprint review* digunakan untuk meninjau hasil yang telah dikerjakan maupun yang belum dikerjakan dan biasa nya dilakukan di akhir. Proses akan selesai jika pekerjaan sudah benar-benar selesai. Proses ini bisa berulang dan pengembang diharuskan cekatan dalam setiap perubahan yang terjadi pada proses pengembangan. Semua proses harus dipahami dan dilaksanakan hal ini menjadi tanggung jawab *scrum master*. Implementasi dapat dilihat pada gambar 3&4 berikut:



Gambar 3. Sprint backlog



Gambar 4. Sprint

b. Alur Kerja Aplikasi

Beberapa alur kerja aplikasi yang akan dikembangkan sebagai berikut:

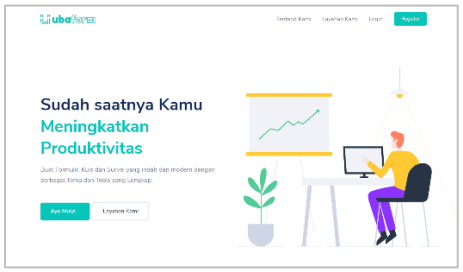
- *User* mengunjungi website ubaform
- *User* akan diarahkan ke halaman beranda atau *home page* dari ubaform
- *User* baru akan melakukan registrasi untuk bisa mengakses layanan ubaform sedangkan *user* yang sudah terdaftar bisa langsung melakukan *login*
- Setelah *login*, *user* akan diarahkan ke halaman *home dashboard.*
- Pada halaman *home dashboard*, *user* dapat langsung membuat formulir, kuis atau survei dengan pilihan menggunakan *template* yang telah tersedia atau membuat dari kanvas.
- Ketika *user* memilih salah satu dari layanan (pembuatan formulir, kuis, dan survei) maka akan diarahkan ke halaman *builder* sesuai pilihan layanan kosong dengan pengaturan tertentu.
- *User* akan dapat melakukan *preview* saat proses pembuatan berlangsung
- *User* akan dapat melakukan kolaborasi dengan *user* lain saat pembuatan dengan membagi *link* atau dengan menambahkan email
- *User* dapat melakukan kustomisasi tampilan *(theme)* secara fleksibel pada saat proses pembuatan
- *User* akan melakukan *submit* setelah selesai dan mendapatkan *link* publikasi untuk digunakan oleh peserta.
- Kemudian peserta dapat mengakses *link* untuk pengerjaan yang telah dibuat
- Peserta akan diarahkan ke tampilan pengerjaan yang telah dibuat sebelumnya
- *User* bisa melihat hasil pengerjaan dari peserta

c. Desain Aplikasi

Selain untuk memudahkan proses pengembangan, desain aplikasi dimaksudkan untuk memaksimalkan pengalaman *user* (*user experience*) dalam menggunakan aplikasi. Berdasarkan perancangan alur kerja aplikasi berikut beberapa tampilan desain aplikasi yang akan dikembangkan:

- *Home Page*

  Berikut tampilan *home page* pada website ubaform dapat dilihat pada gambar 5:

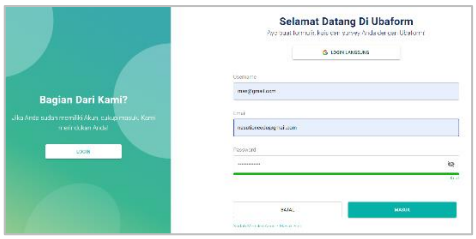


Gambar 5. Gambar home page

- *Authentication Page (Login* dan *Register)*

  Berikut tampilan *authentication page* pada website ubaform dapat dilihat pada gambar 6&7:

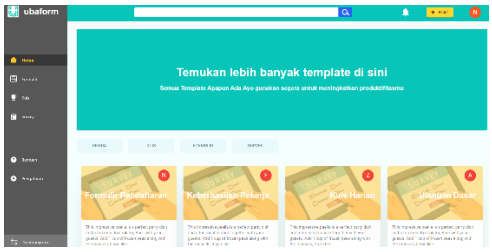
Gambar 6. Login page



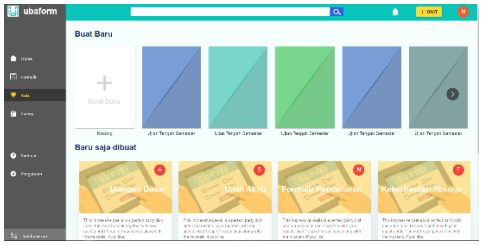
Gambar 7. Register Page

- *Dashboard Page*

  Pada tampilan *dashboard page* digunakan oleh *user* untuk melakukan manajemen baik pembuatan formulir, kuis atau survei. Tampilan ini akan memiliki beberapa list menu pada sidebar. Pada awal nya *user* akan ditampilkan beberapa pilihan *template* yang menarik sesuai dengan kategori dari layanan atau fitur utama. Pada Ketiga layanan menu tersebut akan menampilkan *list* proyek yang telah dibuat sebelumnya serta pilihan pembuatan berdasarkan *template* atau kanvas kosong. Berikut tampilan *dashboard page* pada website ubaform dapat dilihat pada gambar 8 & 9:



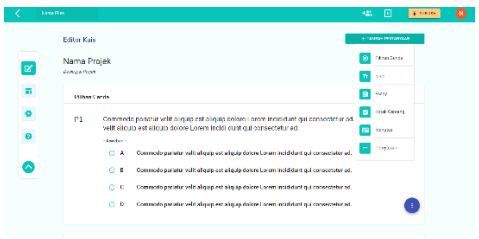
Gambar 8. dashboard home page



Gambar 9. dashboard kuis page

- *Builder Page*

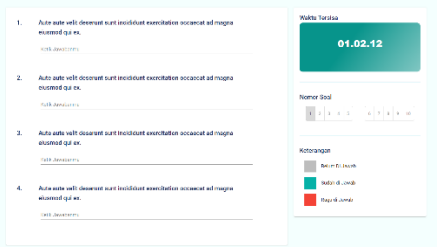
  Pada *builder page* ini akan digunakan sebagai *workspace* pembuatan kuis, formulir dan survei. Berikut tampilan *builder page* pada website ubaform dapat dilihat pada gambar 10:



Gambar 10. builder page

- *Play Page*

  Tampilan *play page* merupakan tampilan formulir, kuis atau survei yang sedang aktif dikerjakan atau diselesaikan oleh peserta secara publik yang mengakses *link* yang dibagikan oleh pembuat satu dari ketiga layanan. Berikut tampilan *dashboard page* pada website ubaform dapat dilihat pada gambar 11:



Gambar 11. play page

d. Teknologi Pengembangan

Pada tahap ini penerapan React JS akan mulai dilakukan untuk mengembangkan tampilan yang telah dibuat sebelumnya.

- Penerapan React JS Pada *home page*

  Pada komponen *header* atau navigasi terdapat perpindahan halaman yang terjadi. Perpindahan dikarenakan *user* mengakses *authentication page (login* dan *register).* Pada dasarnya React JS tidak memiliki komponen pendukung yang mengatur *routing* yang terjadi pada perpindahan tampilan atau halaman dari aplikasi berbasis react. Akan tetapi React JS merupakan *library* javascript yang fleksibel dimana ketika react tidak memiliki komponen yang kompatibel untuk mendukung pengembangan suatu sistem, pada saat itu penggunaan *package* dibutuhkan. Pada implementasi React JS penggunaan *package* sangat umum digunakan oleh pengembang. Selain *package* dapat menghemat waktu pengembangan karena pengembang tidak harus melakukan menuliskan program dari nol juga bersifat *open source*. Untuk dapat menggunakan *package* diperlukan proses instalasi terlebih dahulu dengan menuliskan sintaks **`npm I nama package`** pada terminal. Oleh karena itu untuk dapat berpindah halaman atau tampilan dengan URL diperlukan *package* `react-router-dom`. Artinya ketika *user* ingin mengakses salah satu tampilan maka tidak perlu dilakukan pemuatan ulang halaman. Pada dasarnya ketika CRA *(Create-react-app)* dilakukan kemudian melakukan `npm start` React JS otomatis dijalankan pada *server local* dengan URL `http://localhost:3000/`. Ada beberapa komponen pada *package* `react-router-dom` yang mana masing-masing komponen memiliki fungsionalitas yang

berbeda-beda. Namun pada *home page* akan digunakan beberapa diantaranya, `BrowserRouter, Link dan Route`. `BrowserRouter` digunakan untuk membungkus *root* komponen. Pada *front end* ubaform `BrowserRouter` dikonfigurasikan pada *file* tersendiri. Pada *file* tersebut `BrowserRouter` membungkus beberapa `Route`. Komponen `Route` memiliki parameter `path` dan `component`. `Path` pada `Route` digunakan seperti sebuah objek URL untuk menampilkan komponen tertentu, sedangkan `component` pada `Route` adalah apa yang akan ditampilkan nantinya. Kemudian untuk proses yang terjadi ketika penggunaan `react-router-dom` yaitu navigasi atau *header* dari *home page* memiliki URL untuk mengakses *login page* atau *register page* kemudian dari URL akan diarahkan oleh *package* `react-router-dom` ke komponen `route` untuk kemudian terjadi proses *rendering* pergantian tampilan berdasarkan URL yang diakses.

- Penerapan React JS Pada *login page* dan *register page*

  Proses yang terjadi pada *login page* dan *register page user* akan ditampilkan *form* pengisian email dan *password*. Pada pengisian *field password user* akan dilakukan pengecekan kekuatan kombinasi yang diharapkan dapat meningkatkan pengamanan *password*. Dalam penerapan konsep ini digunakan *package* `zxcvbn`. Pengecekkan kekuatan dari pengisian *password* oleh *user* dengan indikator-indikator khusus. Indikator ini akan mewakili setiap kondisi pengisian *password*. Berikut indikator yang akan dipaparkan pada tabel 1:

Keterangan Kata	**Keterangan Warna**
Tidak ada inputan	Tidak ada Warna
Sangat Lemah	#828282
Lemah	#EA1111
Rentan	#FFAD00
Bagus	#9BC158
Kuat	#00B500

Tabel 1. Indikator password

Proses yang terjadi saat penggunaan *package user* akan melakukan pengisian *password* kemudian oleh *package* setiap pengisian karakter oleh *user* akan diakumulasi sehingga membentuk indikator-indikator pada tabel. Semakin banyak kombinasi karakter yang digunakan oleh *user* akan berbanding lurus dengan kekuatan *password*. Contoh implementasi *package* pada gambar 11 berikut:

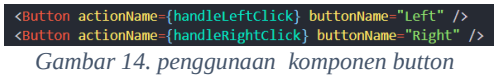
Gambar 12. Indikator password

- Penerapan React JS pada *dashboard page*

  React JS dibuat berbasis komponen yang mana untuk membentuk satu komponen utuh terdiri dari komponen-komponen kecil. Pada *dashboard page* terdapat banyak komponen penyusun. Pembuatan komponen secara terpisah akan memudahkan pengembang dalam melakukan *debugging* serta *maintenance* sistem. *Debugging* merupakan suatu proses untuk menemukan kesalahan *(error)* atau ketidaksesuaian yang terjadi pada suatu program. Biasannya seorang pengembang menghabiskan banyak waktu untuk menemukan suatu *bug*. Tentu hal ini dapat memperlambat proses pengembangan sebuah aplikasi. Namun hal ini dapat diatasi dengan penggunaan React JS. Penggunaan React JS yang bersifat deklaratif memudahkan proses *debugging*. Dalam hal ini diambil kasus pada pengembangan tampilan dashboard *page* komponen `button`. Komponen button dibuat secara terpisah dengan *file* komponen tampilan utama. Komponen button disebut sebagai komponen anak sedangkan komponen *dashboard page* atau komponen utama disebut komponen induk. Berikut contoh komponen button pada gambar 13:

Gambar 13. Komponen button

Komponen dibuat secara deklaratif yang mana fungsi button dideklarasikan dalam variabel `const Button`. Untuk dapat melakukan operan data antara komponen induk dengan komponen anak digunakan props. Data dari props ini berasal dari komponen induk. Berikut contoh komponen induk pada gambar 14:



Gambar 14. penggunaan komponen button

Penggunaan komponen `Button` terjadi dua kali pada gambar 14 namun dengan parameter data yang berbeda. Dua data yang dikirim ke komponen anak antara lain `actionName` dan `buttonName` yang mempresentasikan dua fungsi dan nama yang berbeda. Keunggulan React JS ini dapat dirasakan sangat signifikan ketika komponen yang dibuat sangat besar dan akan selalu digunakan. Kemudahan dan efisiensi pengerjaan yang ditawarkan oleh React JS ini juga dapat terjadi beberapa kendala seperti *bug* dan lain sebagainya. Hal ini bisa saja terjadi pada saat operan data berlangsung atau kesalahan lainnya. Namun React JS sekali lagi menawarkan keunggulan dalam memudahkan pengembang melakukan *debug*. Kembali pada kasus yang sama ketika dua parameter data yang dikirm sebelumnya terjadi *bug* dan hal yang sangat mudah dilakukan adalah menampilkan data operan antara komponen dengan **`console.log`**. Penggunaan **`console.log`** pada kasus ini dapat dilihat gambar 15 berikut:

```
8
9  const Button = (props) => {
10   console.log(props);
11   const { buttonClick, buttonName } = props;
12   return <button onClick={buttonClick}>{buttonName}</button>;
13 };
14
```

Gambar 15 console.log

Pada gambar 15 dua parameter data yang berupa variabel `actionName` dan `buttonName` diinisialisasikan dengan props. Props inilah yang akan ditampilkan dalam *console browser*. Contoh tampilan *console* pada gambar 16 berikut:

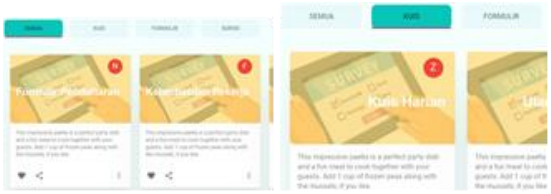
```
▼ {buttonName: "Left", actionName: ƒ} i
  ▼ actionName: () => {…}
      arguments: (...)
      caller: (...)
      length: 0
      name: "handleLeftClick"
    ▶ __proto__: ƒ ()
      [[FunctionLocation]]: App.js:44
    ▶ [[Scopes]]: Scopes[4]
    buttonName: "Left"
  ▶ __proto__: Object
▼ {buttonName: "Right", actionName: ƒ} i
  ▶ actionName: () => {…}
    buttonName: "Right"
  ▶ __proto__: Object
```

Gambar 16 tampilan console browser

Perhatikan gambar 16 terdapat dua button dengan dua data operan berbeda. Dengan dapat melihat sebuah operan data pengembang akan langsung bisa mengetahui letak *bug* yang terjadi dan komponen mana yang harus diperbaiki. Selanjutnya pada pengembangan tampilan *dashboard page* digunakan *package* `syncfusion/ej2-react-navigations` dan `material UI`. Implementasi `material UI` digunakan untuk membuat *header*, *card*, *button* dan komponen lainnya. Berikut implementasi pada gambar 17&18:

```
<BrowserRouter>
  <div className="wrap-tabs-cate">
    <Route
      path="/dashboard/home/"
      render={({ location }) => (
        <Fragment>
          <Tabs className="nav-tabs-cate" value={location.pathname}
            <Tab
              label="Semua"
              value="/dashboard/home/"
              component={NavLink}
              to={allTabs[0]}
              className="nav-tab-title"
            />
```

Gambar 17. Impelementasi material-ui

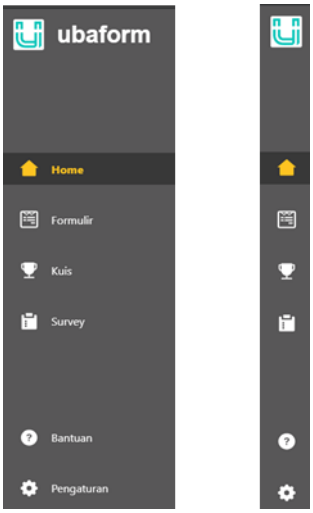


Gambar 18. hasil implementasi material-ui

Sedangkan untuk *package* `syncfusion/ej2-react-navigations` digunakan untuk membuat *sidebar responsive*. Berikut implementasi pada gambar 19&20:

```
62   <SidebarComponent
63     id="dockSidebar"
64     ref={(Sidebar) => (this.dockBar = Sidebar)}
65     enableDock
66     dockSize="62px"
67     width="220px"
68     className="mr-5"
69   >
```

Gambar 19. implementasi syncfusion/ej2-react-navigations



Gambar 20. hasil implementasi syncfusion/ej2-react-navigations

Hasil pembahasan React JS beserta beberapa *package* yang digunakan sudah sesuai dengan apa yang dipaparkan pada tahap perencanaan dan penerapan. Penerapan React JS selain dapat membantu pengembang dalam proses pengembangan dengan bantuan antara lain penggunaan *package* dan penggunaan berbasis komponen yang mempermudah serta mempercepat dalam pengembangan maupun proses *debugging*. Akan tetapi pada saat ini belum ada implementasi terkait pengujian terhadap komponen yang dibuat. Hal ini perlu diimplementasikan karena dikhawatirkan adanya unit komponen yang dibuat memiliki *bug* yang tidak terlihat sehingga merubah data dan sebagainya

V. KESIMPULAN DAN SARAN

Berdasarkan hasil yang didapat dari tahap perencanaan dan implementasi dapat disimpulkan bahwa penerapan React JS dalam pengembangan Front end *startup* ubaform yang berbasis *Single Page Application* (SPA) mampu memberikan kemudahan bagi pengembang dan menghemat waktu karena penggunaan *package* sehingga pengembang tidak perlu melakukan pemrograman setiap komponen dari dari nol. Selain itu React JS datang dengan *Reusable* komponen dan dapat dibuat secara terpisah menjadi beberapa komponen kecil yang dapat membantu pengembang mengetahui letak *error* terjadi sehingga *debugging* menjadi lebih cepat dan dengan memisahkan setiap komponen menjadi unit-unit kecil ketika dilakukan *debug* tidak akan mengganggu komponen lainnya.

Adapun saran untuk meningkatkan pengembangan front end pada *startup* ubaform antara lain diperlukan untuk melakukan *unit testing* pada setiap komponen untuk mengetahui resiko terjadi nya *error* sehingga ketika aplikasi ubaform sudah kompleks nantinya semakin mudah lagi untuk dilakukan *maintenance.*

DAFTAR PUSTAKA

[1] A. D., "Single-page App vs Multi-page Application: What to Choose", Cleveroad Inc. - Web and App development company, 2021. [Online]. Available: https://www.cleveroad.com/blog/single-page-app-vs-multi-page-application-what-to-choose. [Accessed: 21- Jun- 2021].J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2] A. Brothers, "Single Page Application (SPA) vs Multi Page Application (MPA) – Two Development Approaches | ASPER BROTHERS", ASPER BROTHERS, 2019. [Online]. Available: https://asperbrothers.com/blog/spa-vs-mpa/. [Accessed: 21- Jun- 2021].

[3] Y, Team., 2021. Should you hire Angular, AJAX or React Developer? Comparison of technologies.. [online] Youteam.io. Available at: <https://youteam.io/blog/angular-vs-react-comparison/> [Accessed 19 June 2021].

[4] Y. Team, "What Is a Single Page Application (SPA)?", Outsystems.com, 2021. [Online]. Available: https://www.outsystems.com/blog/posts/single-page-application/?utm. [Accessed: 21- Jun- 2021].