

Penerapan *React JS* dan *Axios* untuk Pengembangan *Front-end* Aplikasi *iCare*

Implementation of React JS and Axios for Front-end Development of iCare Application

Sadesty Rahmadhani¹, Mohammad Dyvani Wahyu Wildana², HSP Watulintang Arumdanie³,
Lutfi Hakim^{4*}

Teknologi Rekayasa Perangkat Lunak, Politeknik Negeri Banyuwangi, Banyuwangi, Indonesia^{1,2,3,4}

Email: sadestyrhmadhani012@gmail.com¹, mdyvani@gmail.com², hspwarumd@gmail.com³, lutfi@poliwangi.ac.id⁴

*Corresponding Author

Abstract

PT 3PM IT Solution in Malang City, focuses on project management solution s and software development, involved in developing the iCare application for Astragraphia machine sevice. Initially using Flutter, the company switched to React.js due to performance constraints and technology considerations. This migration follows technological developments and takes advantage of React.js' advantages in building interactive, stateful and reusable UI components. This decision was based on the use of Virtual DOM, high performance, component reusability, easy integration, community support, and a declarative approach. By using Axios as a link between the frontend and backend, it is hoped that application performance will increase. Frontend development is focused on Authentication and History modules. The Authentication module verifies identity, supports session management, and protects user data. The History module makes it easier for users to track and detail consumable history, improving administrative processes and data management. Choosing React.js with a focus on these two modules can increase performance up to 97%.

Keywords: Application performance, React.js, Axios, front-end development

Abstrak

PT 3PM IT Solution, sebuah perusahaan berbasis di Kota Malang, sedang melakukan migrasi *frontend* aplikasi *iCare* dari *Flutter* ke *React.js*. Migrasi ini merupakan bagian dari fokus perusahaan pada solusi manajemen proyek dan pengembangan perangkat lunak. Keputusan untuk beralih ke *React.js* diambil karena kendala performa dan pertimbangan teknologi yang lebih baik. *React.js* dipilih karena keunggulannya dalam membangun komponen UI interaktif, *statefull*, dan *reusable*, serta dukungan yang baik dari komunitas pengembang. *Axios* digunakan sebagai penghubung antara *frontend* dan *backend* yang memiliki dukungan *promises* sehingga mempermudah penanganan permintaan *asynchronous*. Dalam pengerjaannya, tim pengembang dan manajemen proyek PT 3PM IT Solution bertanggung jawab atas keseluruhan proyek migrasi ini. Proses migrasi dilakukan di lokasi perusahaan di Kota Malang dan dilaksanakan selama periode tertentu. Tahap pengujian dilakukan menggunakan fitur *Chrome Lighthouse* untuk memastikan peningkatan performa aplikasi, dan hasilnya menunjukkan peningkatan performa hingga mencapai 97%. Dengan memanfaatkan keunggulan *React.js* serta penggunaan *Axios* sebagai penghubung antara *frontend* dan *backend*, diharapkan aplikasi *iCare* dapat memiliki performa yang lebih baik dan memberikan pengalaman pengguna yang lebih memuaskan kepada pelanggan.

Kata Kunci: Performa aplikasi, *React.js*, *Axios*, pengembangan front-end aplikasi

I. PENDAHULUAN

PT. 3PM IT Solution merupakan salah satu perusahaan yang menyediakan layanan IT berlokasi di Kota Malang [1]. Perusahaan ini terlibat dalam pengembangan aplikasi *iCare* yang mana merupakan aplikasi layanan yang diluncurkan oleh Astragraphia untuk mempermudah dalam melakukan permintaan layanan servis dari mesin-mesin yang dipasarkan oleh Astragraphia. Pada dasarnya, aplikasi *iCare* ini merupakan aplikasi based *web service* dengan *frontend* dan *backend* yang terpisah. *Frontend* adalah bagian dari sistem

yang bertanggung jawab untuk menampilkan *interface* yang digunakan oleh pengguna untuk berinteraksi dengan sistem [2]. Karena terdapat kendala pada performa aplikasi tersebut dan dengan pertimbangan dari teknologi yang digunakan saat itu, dimana teknologi yang digunakan untuk pengembangan *frontend* adalah *Flutter* yang secara dasarnya lebih optimal untuk pengembangan aplikasi *mobile*. Hal tersebut, menjadi faktor utama untuk melakukan migrasi teknologi dari *Flutter* menjadi *React.js*. Beberapa alasan memilih *React.js* selain untuk mengikuti perkembangan teknologi saat ini yaitu karena *React.js* memungkinkan

pengembang dapat membangun sebuah komponen UI yang lebih interaktif, *statefull* dan *reusable* [3] [4]. Selain itu *React.js* juga menggunakan *Virtual DOM* untuk meningkatkan kinerja, kemampuan reusabilitas komponen, mudah diintegrasikan dengan *library* lain, dukungan komunitas yang besar, dan pendekatan deklaratif dalam pembuatan antarmuka pengguna. Dalam mempermudah pembuatan fungsionalitas dari setiap fitur, penulis menggunakan *library Axios* sebagai penghubung antara frontend dan backend. Langkah ini diambil dengan harapan dapat meningkatkan performa keseluruhan aplikasi dan memperbaiki pengalaman pengguna agar lebih optimal.

Seperti penelitian yang sebelumnya pernah dilakukan, Sebuah metode yang dikenal dengan nama *Single Page Application* (SPA) dengan menggunakan *library JavaScript* atau dikenal dengan nama *ReactJS*, dapat dikatakan bisa menanggapi kendala dalam penyajian isi atau konten serta dapat memaksimalkan kinerja dari sebuah *website* [5]. Penelitian sebelumnya juga telah dilakukan oleh Ibnu Choldun dan Reni Rahmadewi dalam penerapan metode *waterfall* pada aplikasi pembelajaran seni budaya berbasis *website* menggunakan *framework React.js*. dalam pembangunan aplikasi ini memanfaatkan metode *waterfall* berserta diagram UML untuk memvisualisasikan model berbasis perangkat lunak. Aplikasi ini dirancang menggunakan bahasa pemrograman *Javascript* dengan *framework React.js*, *Redux*, *Express.js*, *MySQL*, dan *Sequelize*. Berdasarkan pengujian aplikasi tersebut dapat berjalan dengan baik, serta responsif terhadap tampilan [6].

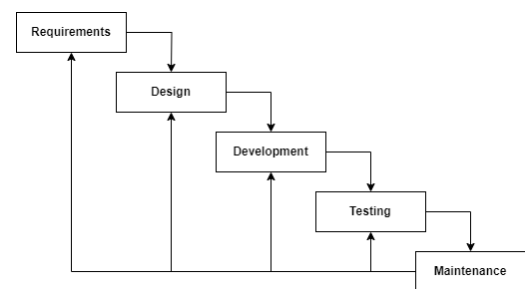
Kemudian pada penelitian dengan judul Pengembangan Sistem Informasi Pengelolaan Persediaan Barang Dengan *ReactJS* Dan *React Native* menggunakan *Prototype*. Penelitian ini saat tahapan implementasi sistem dilakukan dengan menggunakan kerangka kerja *ReactJS* dan *react native*. Setelah implementasi selesai, sistem yang dikembangkan kemudian diuji dengan pengujian *whitebox* untuk pengujian unit. Dari pengujian yang sudah dilakukan menggambarkan bahwa sistem dapat berjalan dengan baik dan sesuai dengan kebutuhan yang telah ditentukan sebelumnya [7]. Penelitian tersebut memiliki keselarasan terhadap penelitian ini, yaitu bertujuan untuk meningkatkan performa aplikasi seperti efisiensi kinerja sistem dan proses operasi aplikasi lebih cepat dengan memanfaatkan *framework ReactJS* dengan hasil pengujian *blackbox* menggunakan pengujian validasi menghasilkan 100% data valid dari 15 kasus uji. Namun, pada aplikasi yang sedang dikembangkan pada penelitian ini memiliki *frontend* dan *backend* yang terpisah sehingga harus menggunakan antarmuka yang dapat menghubungkan antara *frontend* dan *backend*. Dengan memanfaatkan *library Axios* diharapkan dapat mendukung dalam upaya peningkatan performa pada aplikasi *iCare*.

Pada pengembangan *frontend* dari aplikasi ini, terdapat beberapa fitur diantaranya yakni modul Autentikasi, *Request Consumable*, Histori, *Master Data*, Bantuan, dan Beranda. Modul Autentikasi mempunyai 2 peran yang sangat krusial pada aplikasi ini seperti memverifikasi identitas pengguna, mendukung pengelolaan sesi, dan mengamankan data

pribadi pengguna. Tak kalah penting, modul Histori juga sangat membantu pengguna dalam mengingat dan merinci riwayat *consumable* serta memudahkan proses administrasi dan manajemen data. Tidak kalah penting, Fitur-fitur lain memiliki peran yang sangat krusial di dalam aplikasi *iCare* terutama pada proses layanan servis dari mesin yang dipasarkan oleh *astragraphia*.

II. METODE PENELITIAN

Dalam Pengembangan fitur autentikasi dan histori di aplikasi *iCare* dengan metode *waterfall*, tahap dimulai dengan analisis kebutuhan pengguna. Dilanjutkan dengan perancangan sistem yang direpresntasikan dalam bentuk desain. Kemudian, tahap pengembangan melibatkan implementasi fitur sesuai spesifikasi dan pedoman pengembangan, diikuti pengujian fungsionalitas dan keamanan. Setelah fitur dianggap siap, dilakukan pengimplementasian ke lingkungan produksi, dan dilanjutkan dengan pemeliharaan secara rutin untuk memastikan kinerja dan keamanan.



Gambar 1. Metode Waterfall

Metode *Waterfall* merupakan metode yang menyediakan pendekatan alur hidup perangkat lunak secara skensial atau terurut [8]. Tahapan dalam metode ini adalah sebagai berikut:

1. Analisis Kebutuhan (*Requirements*)
Tahap ini adalah tahap pengumpulan kebutuhan dari pengguna dalam perancangan fitur-fitur yang dibutuhkan dalam aplikasi *iCare*. Serta menentukan spesifikasi dasar fitur utama aplikasi seperti autentikasi pengguna, histori servis, permintaan servis, dan data master.
2. Desain (*Design*)
Pada tahap desain dilakukan perancangan struktur letak dan navigasi aplikasi, serta mengidentifikasi komponen-komponen UI yang diperlukan setiap fitur. Kemudian, merancang desain tampilan aplikasi *icare* termasuk antarmuka pengguna, warna, *icon* dan elemen grafis lainnya.
3. Pengembangan (*Development*)
Tim pengembang mulai menerapkan desain ke dalam sebuah kode program menggunakan *React.js* sebagai kerangka kerja utama. Mengembangkan modul-modul yang telah di analisis pada tahap awal.
4. Pengujian (*Testing*)
Setiap komponen dan fungsi diuji secara terpisah

untuk memastikan kebenaran fungsionalitasnya. Serta dilakukan pengujian performa menggunakan salah satu fitur pada *browser* yang sering digunakan pengembang yaitu fitur *lighthouse*.

5. Pendukung atau Pemeliharaan (*Maintenance*)

Tahap terakhir adalah melakukan perbaikan dan penyesuaian jika ada kesalahan atau masalah yang teridentifikasi selama pengujian. Tim juga menyediakan dokumentasi untuk *frontend* aplikasi *iCare*, termasuk panduan pengguna dan spesifikasi teknis.

III. HASIL DAN PEMBAHASAN

A. Gambaran Umum Aplikasi *iCare*

Aplikasi *iCare* adalah inovasi yang Astragraphia hadirkan di masa pandemi untuk memberikan nilai tambah terhadap kualitas layanan perusahaan untuk pelanggan serta mendukung langkah efisiensi perusahaan. *iCare* merupakan sebuah aplikasi berbasis website yang dirancang untuk memfasilitasi proses eskalasi *problem* dan layanan pelanggan Astragraphia dengan mudah. Aplikasi ini memberikan kemudahan bagi pelanggan dalam menyampaikan permasalahan atau kebutuhan layanan, memastikan response cepat dan solusi yang efektif dari tim astragraphia. Dengan *iCare*, pengalaman pelanggan ditingkatkan melalui aksesibilitas yang lebih baik dalam menangani setiap permintaan [9].

B. Teknologi yang Digunakan

Dalam pengembangan perangkat lunak, teknologi mencakup beragam elemen seperti alat, bahasa pemrograman, kerangka kerja (*Framework*), dan *platform* yang berperan dalam seluruh siklus kehidupan perangkat lunak. Dalam proyek ini, beberapa teknologi yang digunakan diantaranya :

1. *JavaScript*

JavaScript adalah bahasa pemrograman yang sering digunakan dalam pengembangan aplikasi web interaktif. Bersama dengan HTML dan CSS, *JavaScript* merupakan salah satu dari tiga teknologi utama dalam pengembangan web. Awalnya, *JavaScript* diciptakan untuk memberikan interaktivitas pada halaman web dengan menjalankan skrip di sisi klien. Ini berarti skrip dijalankan pada perangkat pengguna, seperti *browser* web, memungkinkan manipulasi elemen halaman dan respons terhadap peristiwa tanpa perlu memuat ulang halaman. Seiring waktu, *JavaScript* juga digunakan di sisi server dengan *platform* seperti *Node.js*, memungkinkan pengembang untuk menggunakan *JavaScript* di kedua sisi dan menyediakan integrasi yang lebih baik antara logika di sisi klien dan server dalam pengembangan aplikasi web [10]. *JavaScript* memiliki sintaks yang mirip dengan bahasa pemrograman lain seperti *Java* dan *C*, tetapi memiliki fitur-fitur khusus yang membuatnya unik. Sebagai bahasa yang mendukung paradigma pemrograman

berbasis objek, *JavaScript* memungkinkan pembuatan aplikasi web yang dinamis dengan responsif. Selain itu, banyak *framework JavaScript* populer seperti *React* yang telah dikembangkan 16 untuk mempermudah pengembangan aplikasi web yang akan dibahas pada point selanjutnya[11].

2. *Framework React.js*

React.js adalah sebuah Pustaka (*library*) *JavaScript* yang digunakan untuk membangun antarmuka pengguna (UI) pada aplikasi web. *React* dikembangkan oleh Facebook dan sering digunakan untuk mengembangkan aplikasi web satu halaman (*single-page application*) karena kemampuannya dalam mengelola tampilan yang dinamis [12]. Beberapa poin utama tentang *ReactJS* meliputi penggunaan komponen untuk mengatur UI dengan mudah dan membentuk struktur pohon, konsep *Virtual DOM* untuk meningkatkan kinerja aplikasi dengan membandingkan dan memperbarui bagian-bagian yang berubah, penggunaan *JSX* untuk memungkinkan penulisan kode HTML di dalam *JavaScript*, pemisahan data menjadi *state* dan *props*, aliran data satu arah untuk memudahkan manajemen *state*, dan penggunaan proses rekonsiliasi untuk meminimalkan pengaruh terhadap kinerja dengan hanya memperbarui perubahan yang diperlukan.

3. *Axios*

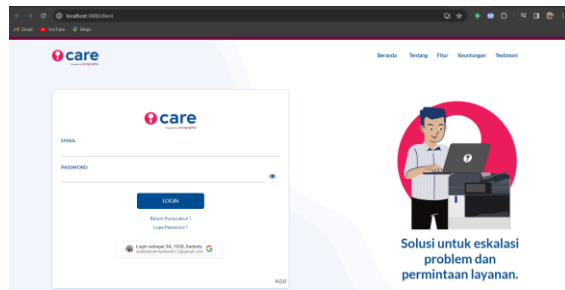
Axios adalah sebuah pustaka (*library*) *JavaScript* yang digunakan untuk melakukan permintaan HTTP dari sisi klien (*client-side*) dalam lingkungan web atau aplikasi [13]. Pustaka ini sering digunakan untuk berkomunikasi dengan API (*Application Programming Interface*) dan mengambil atau mengirim data antara sisi klien dan server. *Axios* menyediakan antarmuka yang bersih dan sederhana untuk mengelola permintaan HTTP serta menangani *response* dengan mudah.

Salah satu fitur utama *Axios* adalah dukungan untuk *Promises*, yang mempermudah penanganan permintaan *asynchronous*. Dengan menggunakan *Promises*, *Axios* memungkinkan pengembang untuk menangani *response* HTTP dengan cara yang lebih mudah dibaca dan dipahami. *Axios* juga menyediakan berbagai fitur lainnya, seperti dukungan untuk pembatalan permintaan, penanganan otentikasi, dan intersepsi permintaan atau respons untuk manipulasi lebih lanjut sebelum atau setelah proses pengiriman.

C. Penerapan *Framework ReactJS* pada Aplikasi *iCare*

Pada tahap ini penerapan *ReactJS* akan mulai dilakukan untuk mengembangkan tampilan yang telah dibuat sebelumnya. Berikut contoh dari beberapa fitur yang

1. Penerapan *ReactJS* pada halaman *login*

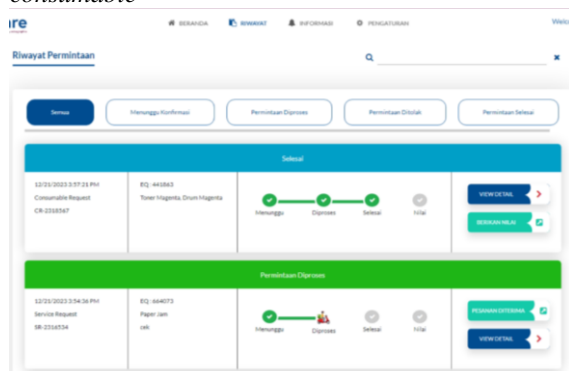


Gambar 2. Tampilan Login

Fitur login pada aplikasi *iCare* memberikan akses eksklusif kepada pengguna, memungkinkan masuk dan menjelajahi fitur-fitur yang terdapat dalam aplikasi. Ketika pengguna melakukan *login* secara manual dengan memasukkan *email* dan *password*, sistem memanggil fungsi *useLogin()*. Fungsi ini tidak hanya bertugas melakukan validasi terhadap masukan pengguna, tetapi juga mempersiapkan langkah-langkah sebelum mengirim permintaan ke *backend* menggunakan *Axios*. Dalam persiapan tersebut, parameter *type* dengan nilai "normal" disertakan untuk memberikan informasi tambahan kepada *backend* yang diperlukan untuk pengecekan lebih lanjut.

Sebaliknya, apabila pengguna memilih untuk *login* menggunakan akun *Google* yang sudah terdaftar, sistem akan menjalankan fungsi *googleLogin()*. Fungsi ini dirancang khusus untuk menangani jenis login melalui *Google* dan menyertakan parameter *type* dengan nilai "google". Setelah menerima *response* dari *backend*, sistem akan mengevaluasi keberhasilan *response* tersebut. Jika *response* berhasil, data *login* yang diterima akan disimpan dalam *local storage* sebagai sesi *login* pengguna. Namun, jika *response* gagal, sistem akan menampilkan pesan yang relevan berdasarkan informasi yang diberikan oleh *backend*, memberikan pengguna umpan balik yang jelas terkait masalah yang mungkin terjadi pada saat *login*.

2. Penerapan *React.js* pada halaman riwayat *consumable*

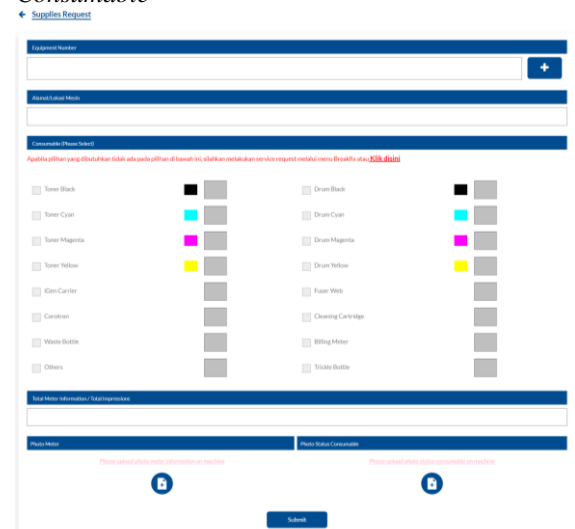


Gambar 3. Halaman Riwayat Permintaan

Fitur riwayat *consumable* pada aplikasi *iCare* adalah

layanan yang memberikan pengguna kemampuan untuk memeriksa secara rinci permintaan serta melacak *consumable* dan eskalasi masalah yang telah dikirim. Ketika pengguna mengakses fitur riwayat, sistem akan menjalankan fungsi *init()* untuk memulai proses pengambilan data riwayat dari *backend* menggunakan *Axios*. Dalam tahapan ini, sistem menyertakan parameter status, skip, dan *rowCount* agar dapat melakukan permintaan *get data* secara spesifik. Setelah mendapatkan *response* dari *backend*, dilakukan pengecekan untuk memastikan keberhasilan permintaan tersebut. Jika permintaan berhasil, sistem akan melanjutkan dengan melakukan pemetaan data yang diterima sebelum menampilkan informasi riwayat tersebut. Sebaliknya, jika permintaan gagal, sistem akan merespon dengan menampilkan pesan sesuai dengan informasi *response* yang diberikan oleh *backend*. Pendekatan ini memberikan pengguna pengalaman yang konsisten dan memberikan tanggapan yang informatif dalam setiap situasi, baik berhasil maupun tidak.

3. Penerapan *ReactJS* pada halaman *Request Consumable*

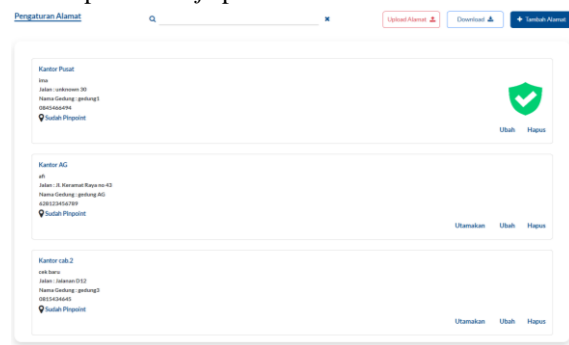


Gambar 4 Halaman Request Consumable

Request consumable (*Supplies Request*) dalam Aplikasi *iCare* Astragraphia memberikan kemudahan bagi pengguna dalam mengelola kebutuhan bahan pakai dan suku cadang mesin printer. Melalui fitur ini, pengguna dapat dengan mudah mengajukan permintaan untuk mendapatkan *consumable* seperti kertas, toner, atau tinta printer, serta suku cadang tertentu yang diperlukan. *Consumable* yang ditampilkan pada halaman ini mengambil data dari API yang telah disediakan dengan menjalankan fungsi *masterRequest()*. Sehingga admin dari Astragraphia dapat dengan mudah mengatur *consumable* apa saja yang tersedia. Begitupun dengan pelanggan, dapat melakukan *request consumable*

dengan sangat mudah hanya dengan mengisi formulir yang tersedia pada halaman tersebut. Kemudian, hasil dari inputan yang diinputkan oleh *user* akan ditampung pada sebuah *state* yang selanjutnya dikirimkan ke *backend* untuk proses lebih lanjut. Sedangkan *state* adalah sebuah objek yang berisi data yang akan digunakan oleh komponen namun *state* bersifat *read-write* yang artinya *developer* dapat mengubah *state* tersebut [14].

4. Penerapan *React.js* pada halaman Daftar Alamat



Gambar 5 Halaman Daftar Alamat

Daftar Alamat ini sangat berguna pada saat *user* atau pelanggan melakukan *request consumable*. Oleh karena itu, sebelum melakukan *request* pelanggan diharuskan melengkapi data alamat mereka melalui formulir yang telah disediakan dengan menekan tombol “tambah alamat”. Setelah alamat yang diinputkan berhasil disimpan ke *backend* maka akan ditampilkan pada daftar alamat seperti pada gambar 5 dengan menjalankan fungsi *init()*. Pada fitur ini memiliki beberapa fungsi seperti ubah, hapus dan utamakan. Setiap fungsi memiliki kegunaan masing-masing, misalnya fungsi utamakan yang digunakan untuk mengutamakan alamat yang sering dipakai. Fungsi ini dijalankan dengan menggunakan API untuk mengirimkan data ke *backend*. Sehingga pada saat pelanggan ingin melakukan *request* akan otomatis menggunakan alamat yang telah diutamakan.

D. Penerapan *Library Axios* pada Aplikasi *iCare*

Library Axios digunakan untuk melakukan pengiriman permintaan HTTP di lingkungan *JavaScript*, khususnya pada pengembangan aplikasi web (Rawat and Mahajan 2020). Dalam konteks pengembangan *frontend* aplikasi *iCare* menggunakan *framework React* di PT 3PM IT Solution, *Axios* dapat digunakan untuk berinteraksi dengan *backend server*, mengambil dan mengirim data antara *frontend* dan *backend*. Sebelum *Axios* dapat dijalankan perlu dilakukan

instalasi dan impor terlebih dahulu dengan perintah “*npm install axios*” dan “*import axios from 'axios'*”.

Untuk menghubungkan antara *frontend* dengan *backend* menggunakan *Axios*, memiliki struktur yang hampir sama dengan antarmuka lain yaitu membuat sebuah fungsi dengan menggunakan metode *GET* atau *POST* sesuai dengan kebutuhan. Metode *GET* dan *POST* memiliki perbedaan mendasar, di mana metode *GET* melakukan parsing data pada URL, sedangkan metode *POST* melakukan *parsing query string* pada *background* [15]. Contoh penggunaan untuk membuat permintaan *GET* dan *POST*:

```
export const getRiwayatOrderByStatus = async (req) => {
  createAuthRefreshInterceptor(axios, refreshAuthLogic);
  const url = `${appConfig.BASE_API}/request/readByStatus`;
  try {
    const res = await axios.post(url, req, jsonHeaderAuth());
    return {status: res.status, data: res.data}
  } catch (error) {
    return { status: error.code, message: error.message }
  }
};

export const getDetailRiwayatOrder = async (id) => {
  createAuthRefreshInterceptor(axios, refreshAuthLogic);
  try {
    const res = await
    axios.get(`${appConfig.BASE_API}/request/readdetail/${id}`,
    jsonHeaderAuth())

    return {status: res.status, data: res.data}
  } catch (error) {
    throw error
  }
};
```

Gambar 6. Sintaks Permintaan *GET* dan *POST* pada *Axios*

Dalam konteks pengembangan *frontend React*, *Axios* dapat digunakan di komponen-komponen untuk menangani permintaan data atau interaksi dengan *backend*. Misalnya, dalam kasus aplikasi *iCare*, permintaan untuk memperoleh atau menyimpan data pada beberapa fitur salah satunya fitur Autentikasi. Contoh penerapan permintaan data menggunakan *Axios* pada Komponen *React*:

```
const res = await resetPassword({userid: location.state.userid,
otp: location.state.otp, newpassword: newPassword})
if(res.status == 200 && res.data != null) {
  setShowPopup(true)
  setAlertOption({title: '', message: "Sukses ubah
password", redirect: true, url: '/'})
} else {
  setShowPopup(true)
  setAlertOption({title: 'Error', message: 'Opps!
terjadi kesalahan', redirect: true, url: '/'})
}
```

Gambar 7. Pemanggilan Fungsi yang Berisi Permintaan HTTP Menggunakan *Axios* pada Komponen *React*

Penerapan *Library Axios* ini membantu dalam mempermudah pengembangan, memungkinkan komunikasi yang efisien antara *frontend* dan *backend*, serta menyediakan mekanisme yang handal untuk menangani permintaan HTTP dalam aplikasi *iCare*.

E. Perbandingan Penggunaan *Library Axios* dan *fetch()*

Berdasarkan penelitian ini terdapat keunggulan dari *Library Axios* yang memungkinkan beberapa pengembang lebih memilihnya dari permintaan HTTP bawaan karena kemudahan dalam penggunaannya, diantaranya:

- *Backward Compability*

Menggunakan *Axios* memiliki dukungan *browser* yang luas. Bahkan *browser* lama seperti IE11 dapat menjalankan *Axios* tanpa masalah apa pun. Hal ini karena digunakan *XMLHttpRequest*. Sedangkan menggunakan permintaan HTTP bawaan seperti *fetch()* hanya mendukung Chrome 42+, Firefox 39+, Edge 14+, dan Safari 10.3+.

- **Response Timeout**

Di *Axios*, dapat menggunakan properti opsional di objek konfigurasi untuk menyetel jumlah milidetik sebelum permintaan dibatalkan. Contoh kode penanganan *response timeout* pada *Axios*.

```
axios().catch(error => {
  if (error.code === 'ECONNABORTED') {
    console.log('Request timed out');
  } else {
    console.log(error.message);
  }
});
```

Gambar 8. Kode Penanganan *Response Timeout* pada *Axios*

Pada *fetch()* menyediakan fungsionalitas serupa, namun tidak sesederhana versi *Axios*.

- **Transformasi data JSON otomatis**

Axios secara otomatis merangkai data saat mengirimkan permintaan, seperti yang terdapat pada gambag di bawah.

```
const res = await axios.post(url, payload,
  {
    headers: {
      'Content-Type': 'application/json',
    },
  }
);
```

Gambar 9. Kode Program Transformasi Data JSON Otomatis sedangkan menggunakan *fetch()* pengembang perlu melakukannya secara manual dengan memasukkan kode program di bawah.

```
const res = await fetch(url, {
  method: 'POST',
  body: JSON.stringify(payload)
});
```

Gambar 10. Kode Program Transformasi Data JSON Manual dengan *Fetch()*

- **HTTP Interceptor**

Salah satu keunggulan *Axios* adalah kemampuannya untuk mencegat permintaan HTTP. *HTTP interceptor* berguna ketika pengembang perlu memeriksa atau mengubah permintaan HTTP dari aplikasi ke server atau sebaliknya.

```
createAuthRefreshInterceptor(axios, refreshAuthLogic)
```

Gambar 11 Kode program *HTTP Interceptor* pada *Axios*

F. Pengujian Performa Aplikasi iCare

Migrasi teknologi dari *Flutter* ke *ReactJS* merupakan usaha yang dilakukan untuk meningkatkan performa aplikasi *iCare*. Seperti pada pengujian yang telah penulis lakukan, perbedaan performa dari aplikasi *iCare* dengan menggunakan teknologi *Flutter* dan *ReactJS*, menampilkan hasil yang cukup signifikan. Berikut adalah hasil pengujian yang penulis lakukan pada aplikasi *iCare* dengan menggunakan teknologi *Flutter*. berdasarkan hasil pengujian terlihat tingkat kualitas performa keseluruhan dari aplikasi *iCare* berada di bawah angka 50%. Hal ini cukup

berpengaruh pada kenyamanan pengguna dan menjadikan aplikasi terasa sangat berat untuk dijalankan. Tidak hanya itu, pada aplikasi *iCare* yang menggunakan teknologi *Flutter*, juga mempunyai tingkat kualitas SEO (*Search Engine Optimization*) pada angka 83% yang menunjukkan bahwa aplikasi *iCare* memiliki visibilitas yang rendah pada hasil pencarian. Dengan hasil tersebut, membuat aplikasi *iCare* tidak dapat memaksimalkan potensi bisnis secara online.

Berbeda dengan hasil pengujian pada aplikasi *iCare* yang menggunakan teknologi *ReactJS*, hasil pengujian menunjukkan hasil yang cukup signifikan. Setelah melakukan pengujian aplikasi *iCare* yang menggunakan teknologi *ReactJS*, menunjukkan tingkat kualitas performa pada angka 63%. Hasil ini bukanlah hasil final, terlebih masih terdapat tahap optimalisasi kode sebelum akhirnya dilakukan *deployment* [16]. Selain itu, dengan tingkat kualitas aksesibilitas 93%, menunjukkan bahwa komitmen terhadap penggunaan praktik desain yang baik dan memastikan bahwa pengguna dengan berbagai kemampuan dapat mengakses dan menggunakan aplikasi dengan mudah. *Best practice* dengan tingkat kualitas pada angka 91% menunjukkan implementasi standar pengembangan dengan kepatuhan terhadap pedoman dan metode terbaik dalam pengembangan aplikasi. Sementara itu, dengan tingkat kualitas SEO sebesar 96%, menunjukkan optimasi yang kuat untuk mesin pencari dan memastikan visibilitas *online* yang optimal, dengan begitu membuka potensi aplikasi dari segi bisnis secara *online*.

Namun, pengujian ini dipengaruhi oleh *chace* yang tercatat pada *browser* setiap kali mengakses aplikasi tersebut. Oleh karena itu, diperlukan sebuah tabel hasil untuk melakukan pengujian beberapa kali guna memastikan keakuratan peningkatan performa pada aplikasi *iCare* setelah dilakukan migrasi ke *ReactJS*. Dari hasil percobaan 1 sampai dengan percobaan 5 dapat dikalkulasikan guna mendapatkan hasil rata-rata presentase performa dari sistem menggunakan *Flutter* dan menggunakan *React.js* dengan persamaan berikut.

$$\text{rata-rata performa} = \frac{\text{jumlah nilai performance seluruh percobaan}}{\text{jumlah percobaan}} \times 100 \quad (1)$$

Berdasarkan persamaan (1) teknologi *Flutter* mendapatkan hasil nilai presentase rata-rata performa 45,6%, sedangkan *React.js* 87,6% yang menunjukkan bahwa *React.js* memiliki performa yang lebih unggul dibanding dengan *Flutter*.

Tabel 1. Daftar Pengujian Performa Aplikasi *iCare*

No	Tes	Hasil	
		Flutter	React
1	Percobaan 1	Performance: 33% Accessibility: 77% Best Practice: 95% SEO: 83%	Performance: 63% Accessibility: 93% Best Practice: 91% SEO: 96%

2	Percobaan 2	Performance: 62% Accessibility: 77% Best Practice: 95% SEO: 80%	Performance: 97% Accessibility: 93% Best Practice: 91% SEO: 100%
3	Percobaan 3	Performance: 38% Accessibility: 78% Best Practice: 95% SEO: 80%	Performance: 95% Accessibility: 93% Best Practice: 91% SEO: 100%
4	Percobaan 4	Performance: 41% Accessibility: 78% Best Practice: 95% SEO: 80%	Performance: 97% Accessibility: 93% Best Practice: 9% SEO: 100%
5	Percobaan 5	Performance: 54% Accessibility: 78% Best Practice: 95% SEO: 80%	Performance: 86% Accessibility: 93% Best Practice: 91% SEO: 100%

Setelah memastikan bahwa aplikasi berjalan dengan baik dan memenuhi standar performa yang diinginkan, langkah selanjutnya adalah merilis aplikasi *iCare* kepada pengguna. Dengan demikian, tahapan ini bukan hanya sebagai evaluasi kualitas teknis, tetapi juga sebagai persiapan krusial sebelum aplikasi dapat dinikmati oleh pengguna secara luas.

IV. KESIMPULAN

Hasil dan pembahasan Manajemen *Knowledge Infrastructure* (MKI) di PT 3PM IT Solution berhasil mengatasi beberapa masalah yang diidentifikasi. Migrasi ke *React.js* memberikan dampak positif pada performa dan kualitas aplikasi *iCare*. Implementasi fitur Autentikasi dan Histori meningkatkan pengalaman pengguna dengan keamanan, efisiensi, dan informasi optimal. Penggunaan *library Axios* dalam pengelolaan permintaan HTTP memberikan keunggulan dibandingkan dengan *fetch()*. Hasil pengujian performa menunjukkan kualitas aplikasi *iCare* dengan *React.js* lebih baik daripada *Flutter*, dengan tingkat kualitas performa di atas 60%. Keseluruhan, implementasi MKI memberikan solusi komprehensif untuk meningkatkan pengembangan dan pengelolaan aplikasi *iCare*.

Kesimpulan menggambarkan jawaban dari hipotesis dan/atau tujuan penelitian atau temuan ilmiah yang diperoleh. Kesimpulan bukan berisi perulangan dari hasil dan pembahasan, tetapi lebih kepada ringkasan hasil temuan seperti yang diharapkan di tujuan atau hipotesis. Bila perlu, di bagian akhir kesimpulan dapat juga dituliskan hal-hal yang akan dilakukan terkait dengan gagasan selanjutnya dari penelitian tersebut.

REFERENSI

- [1] 3PM IT Solution, 3PM IT Solution, 3PM IT Solution (2024). <https://www.3pm-solution.com/profile> (accessed March 26, 2024).
- [2] Terttiavini, I Made Agus Oka Gunawan, Kraugusteeliana, E. Winarno, Rony Sandra Yofa Zebua, Perancangan dan Implementasi Frontend Web untuk Sistem Pengaduan Masyarakat, *Jurnal Informasi Dan Teknologi* (2023) 112–126. <https://doi.org/10.37034/jidt.v5i1.290>.
- [3] J. Panjaitan, A.F. Pakpahan, Perancangan Sistem E-Reporting Menggunakan ReactJS dan Firebase, *Jurnal Teknik Informatika*

- Dan Sistem Informasi 7 (2021). <https://doi.org/10.28932/jutisi.v7i1.3098>.
- [4] A. Wibowo, M.M. Zain, Pemanfaatan ReactJS dan Protokol MQTT untuk Visualisasi Sinyal Lampu dan Notifikasi secara Waktu Nyata pada Sistem Pemonitor APILL di Kota Pekanbaru, *Jurnal Komputer Terapan* 7 (2021) 314–328. <https://doi.org/10.35143/jkt.v7i2.5108>.
- [5] M.F. Santoso, TEKNIK SINGLE PAGE APPLICATION (SPA) LAYOUT WEB DENGAN MENGGUNAKAN REACT JS DAN BOOTSTRAP, *Jurnal Khatulistiwa Informatika* 9 (2021). <https://doi.org/10.31294/jki.v9i2.11357>.
- [6] I. Choldun, R. Rahmadewi, Penerapan Metode Waterfall Pada Aplikasi Pembelajaran Seni Budaya Berbasis Website Menggunakan Framework Reactjs, *Jurnal Ilmiah Wahana Pendidikan*, Juli 2023 (n.d.) 335–348. <https://doi.org/10.5281/zenodo.8151254>.
- [7] F.F. Nursaid, A.H. Brata, A.P. Kharisma, Pengembangan Sistem Informasi Pengelolaan Persediaan Barang Dengan ReactJS Dan React Native Menggunakan Prototype (Studi Kasus : Toko Uda Fajri), *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer* 4 (2020).
- [8] M. Badrul, Penerapan Metode waterfall untuk Perancangan Sistem Informasi Inventory Pada Toko Keramik Bintang Terang, *PROSISKO: Jurnal Pengembangan Riset Dan Observasi Sistem Komputer* 8 (2021) 57–52. <https://doi.org/10.30656/prosisko.v8i2.3852>.
- [9] Astragraphia, *iCare by Astragraphia*, Documentsolution (2024). <https://www.astragraphia.co.id/id/keberlanjutan/produk-dan-bisnis/portfolio-roadmap/tanggung-jawab-produk-dan-layanan> (accessed March 26, 2024).
- [10] A. Noviantoro, A.B. Silviana, R.R. Fitriani, H.P. Permatasari, RANCANGAN DAN IMPLEMENTASI APLIKASI SEWA LAPANGAN BADMINTON WILAYAH DEPOK BERBASIS WEB, *Jurnal Teknik Dan Science* 1 (2022) 88–103. <https://doi.org/10.56127/jts.v1i2.108>.
- [11] V. Siahaan, R.H. Sianipar, JavaScript : Dari A Sampai Z, Sparta Publisher (2018).
- [12] N.T. Syah Lampah, E.B. Setiawan, Aplikasi Asesmen Anak Berkebutuhan Khusus di SLB Rafaha Arjasari Menggunakan Progressive Web App, *ULTIMA Computing* 10 (2019) 65–74. <https://doi.org/10.31937/sk.v10i2.958>.
- [13] B. Angga, 5 Library React JS Untuk Web Developer, Buildwithangga (2024).
- [14] K. Juan, S. Budi, Pengembangan Menu Digital Menggunakan ReactJs Implementasi Hasil Belajar Studi Independen di Frontend Engineering Program Ruangguru CAMP (Career Acceleration Bootcamp), *Jurnal Strategi* 5 (2023).
- [15] I.P.A. Eka Pratama, Pengujian Performansi Lima Back-End JavaScript Framework Menggunakan Metode GET dan POST, *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)* 4 (2020). <https://doi.org/10.29207/resti.v4i6.2675>.
- [16] Herman, A. Geovany, Analisis Rendering Performa Antara Serverside dan Client Side pada Web Application, *Jurnal Ilmiah Betrik* (2022). <https://doi.org/https://doi.org/10.36050/betrik.v13i03> DESEMBER.38.