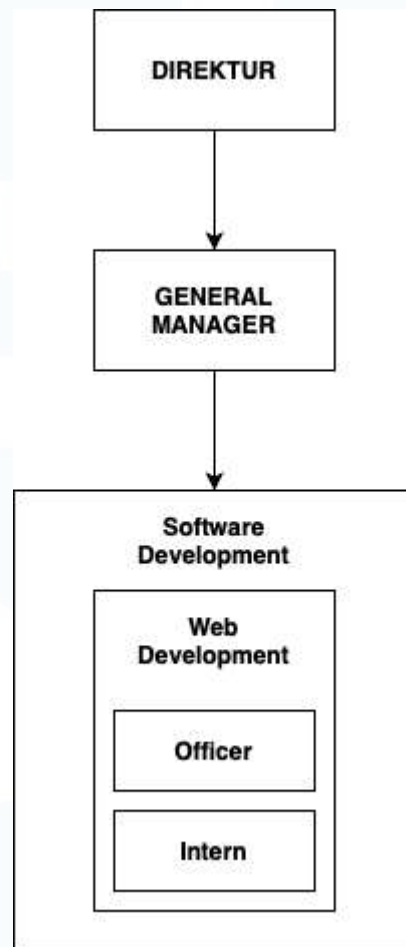


BAB III

PELAKSANAAN KERJA MAGANG

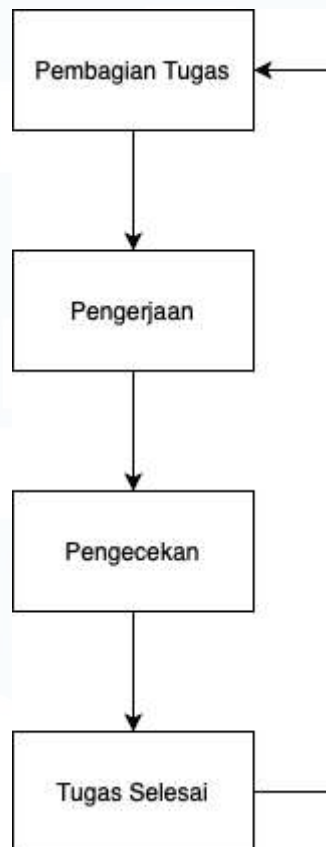
3.1 Kedudukan dan Koordinasi



Gambar 3.1 Struktur Divisi *Software Development*

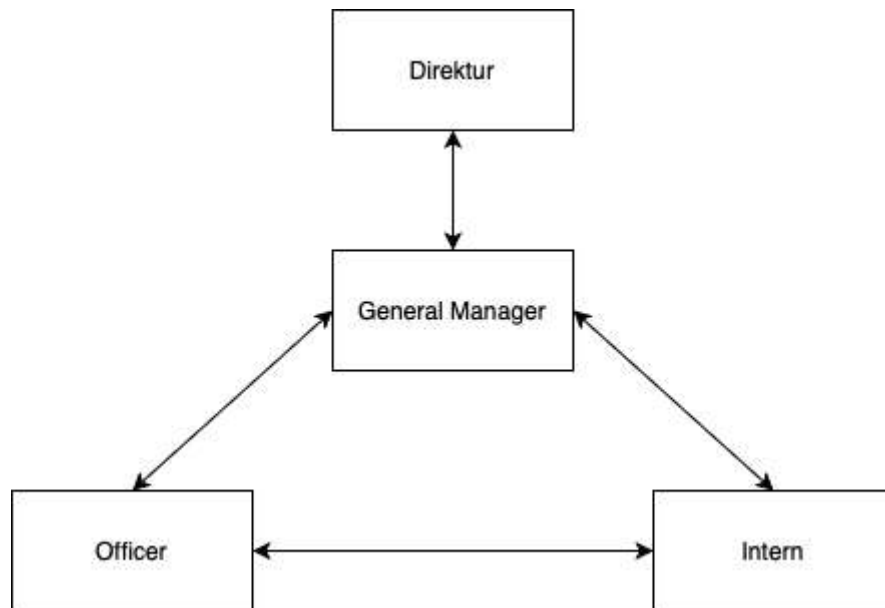
Web developer intern berada pada divisi *Software Development*, tepatnya pada departemen *Web Development*. Divisi *Software Development* berada dibawah tanggung jawab *general manager*. Pekerja magang divisi web developer dibimbing oleh general manager sebagai supervisor pekerja magang selama melaksanakan praktek kerja magang. Supervisor mengawasi pekerja magang

dalam keseharian pekerja magang dan memberikan *approval* pada *platform* Kampus Merdeka selama kerja magang berlangsung.



Gambar 3.2 Alur Kerja Divisi *Software Development*

Dalam proyek pembuatan *website company profile* dan layanan perusahaan, tugas diberikan dengan menggunakan aplikasi WhatsApp. Begitu juga apabila tugas sudah selesai dikerjakan maka pekerja magang akan melaporkan hasil pekerjaannya kepada *supervisor* untuk dilakukan pengecekan, apabila tidak ada revisi maka *supervisor* akan melaporkan kembali kepada pekerja magang.



Gambar 3.3 Alur Koordinasi Divisi *Software Development*

Semua proyek atau pekerjaan yang akan dikerjakan akan disampaikan direktur kepada *general manager*, kemudian *general manager* akan menyampaikan kepada karyawan lain termasuk pekerja magang. dalam pengerjaan proyek atau pekerjaan, apabila pekerja magang menemukan kesulitan atau kendala, maka pekerja magang dapat melakukan diskusi dengan karyawan lain secara langsung atau melalui *chat* via WhatsApp. Selain itu terdapat meeting yang dilakukan oleh divisi untuk mengetahui progress dari proyek yang sedang dikerjakan.

3.2 Tugas dan Uraian Kerja Magang

Web developer intern pada PT Solusi Usaha Berdikari memiliki tugas yaitu membantu tim *Software Development* dalam membuat dan mengembangkan sebuah *website company profile*, *website* tersebut berisikan *profile* dari perusahaan serta layanan-layanan yang ditawarkan oleh perusahaan kepada publik. Dalam mengerjakan pekerjaan yang telah diberikan pekerja magang dibimbing oleh officer yang berada dalam satu divisi. *Website company profile* ini dibangun dengan menggunakan *framework* ReactJS, dan Bootstrap 5 untuk CSS

framework-nya. Untuk detail pekerjaan yang dilakukan oleh pekerja magang pada PT Solusi Usaha Berdikari, dapat dilihat pada tabel 3.1 berikut.

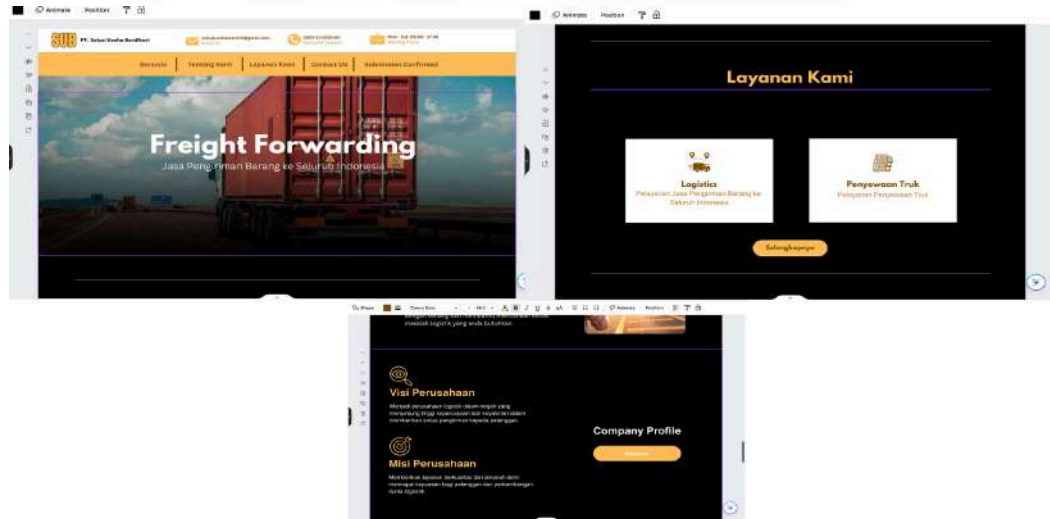
Tabel 3.1 *Timeline* Kerja Magang

No.	Minggu ke-	Pekerjaan
1	1	Membuat <i>Design Website</i>
2	2-3	Belajar React.js
3	4-5	Membuat <i>Route, Homepage, Navbar, dan Footer Website</i>
4	6	Membuat <i>Page Tentang Kami</i>
5	7	Membuat <i>Page Layanan Kami</i>
6	8	Membuat <i>Page Contact Us</i>
7	9	Membuat <i>Map Direction</i>
8	10	Membuat <i>Website Responsive</i>
9	11-12	<i>Hosting Website</i>

3.2.1 Membuat *Design Website*

Kegiatan pekerja magang pada minggu pertama adalah membuat design website. Design website dibuat dengan menggunakan aplikasi design Canva, pertama-tama pekerja magang menentukan color palette sebagai warna dasar pada website untuk diajukan kepada pembimbing lapangan atau supervisor. Setelah pengajuan color palette diajukan dan diterima, pekerja magang melanjutkan pekerjaan dengan membuat design website secara keseluruhan dan mengajukan kembali hasil design

pada supervisor untuk asistensi. Gambar 3.4 adalah beberapa hasil tangkapan layar dari design website yang dibuat pada aplikasi canva.



Gambar 3.4 Design Website di Aplikasi Canva

3.2.2 Belajar *Framework React.js*

Setelah membuat design *website* selesai, pekerja magang diminta untuk mempelajari *React.js*, khususnya *React.js version 18.3.1*. pekerja magang diminta untuk mempelajari dokumentasi dari *React.js*, dan bagaimana cara menerapkannya dalam sebuah *project*. *React.js* atau biasa dikenal dengan *React* adalah sebuah *library JavaScript* yang digunakan untuk membangun user *interface* (UI) dalam aplikasi *website*. *React* dikembangkan oleh facebook, *React* juga memberikan cara yang sangat efisien untuk mengelola visual aplikasi dengan memperbarui dan merender komponen UI secara dinamis ketika data berubah.

Pada saat mempelajari *React*, pekerja magang mempelajari bagaimana cara menyusun komponen, menambahkan *markup* dan *style*, menampilkan data, merender, merespon *event* dan *update*

screen, serta bagaimana cara berbagi data antar komponen. Selain itu dalam *React* terdapat beberapa konsep kunci yang terkait dengan *React.js*. beberapa konsep kunci antara lain sebagai berikut:

a. Komponen

React memisahkan UI menjadi komponen-komponen kecil dan mandiri. Komponen-komponen ini dapat diatur secara hierarki untuk membangun visual yang kompleks. Setiap Komponen pada *React* memiliki logika dan strukturnya sendiri.

b. *Virtual DOM*

React menggunakan konsep *Virtual DOM (Document Object Model)* untuk meningkatkan kinerja aplikasi. *Virtual DOM* adalah representasi ringan dari *DOM* aktual yang disimpan di dalam memori. Apabila terdapat perubahan pada data, *React* memperbarui *Virtual DOM* terlebih dahulu dan membandingkannya dengan *DOM* yang ada untuk menentukan perubahan yang sebenarnya perlu diterapkan pada halaman.

c. JSX

React menggunakan *JSX (JavaScript XML)* sebagai eksistensi sintaks untuk menulis struktur visual dalam *JavaScript*. *JSX* juga memungkinkan pengembang menyatukan *HTML* code dan *JavaScript* dalam satu *file*, yang membuatnya lebih mudah untuk memahami dan memanipulasi visual.

d. *State dan Props*

React Memungkinkan penggunaan *state* dan *props* untuk mengelola data di dalam komponen. *State* digunakan untuk data yang dikelola secara lokal oleh komponen tersebut, sementara

props digunakan untuk mentransfer data dari komponen yang satu ke komponen yang lain.

e. Reconciliations

React Menyediakan algoritma reconciliations yang efisien untuk memastikan perubahan pada virtual DOM diterapkan ke DOM aktual secara optimal. Hal ini membantu dalam mempertahankan kinerja aplikasi yang responsif, terutama pada aplikasi yang kompleks.

Seiring berkembangnya zaman, *React* menjadi semakin populer dalam pengembangan *website* modern karena sifatnya yang deklaratif, performa yang baik, dan komunitas pengembang yang besar.

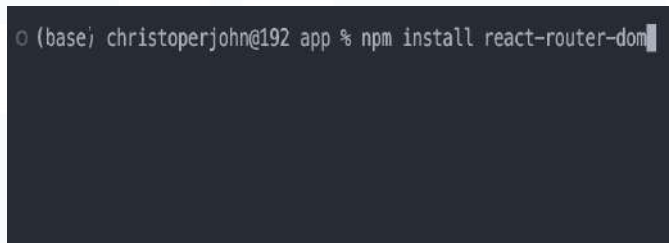
3.2.3 Membuat *Route*, *Homepage*, *Navbar* dan *Footer Website*

Pada Tahap ini, pekerja magang mulai mengerjakan proyek yang diberikan yaitu membuat sebuah *website company profile* dan layanan-layanan yang ditawarkan kepada konsumen. Pertama-tama, pekerja magang menginstal *React* pada proyek yang sedang dikerjakan, kemudian dilanjutkan dengan membuat *route* antar *page* pada *website*. Pembuatan *route* pada *React* melibatkan penggunaan *React Router*, sebuah *library* yang digunakan untuk mengatur navigasi dalam aplikasi *Website React*. Dengan *React Router*, dengan begitu dapat menentukan bagaimana komponen-komponen *React* ditampilkan berdasarkan URL yang saat ini dibuka oleh pengguna.

Untuk membuat *route* pada *React*, maka diperlukan *React*

Router. Berikut adalah langkah-langkah yang umum digunakan untuk membuat *route* pada *React*:

a. *Install React Router*



```
(base) christoperjohn@192 app % npm install react-router-dom
```

Gambar 3.5 *Install React Router*

Langkah pertama yang harus dilakukan adalah menginstal *React router*. Instalasi dapat dilakukan pada terminal dengan code **npm install react-router-dom**. Contohnya dapat dilihat pada gambar 3.5.

b. *Mengatur Route*



```
import { BrowserRouter, Routes, Route, useParams } from "react-router-dom";
```

Gambar 3.6 *Import React Router*

Untuk mengatur *route* harus dilakukan pada komponen atau *file* induk yaitu **App.js**. di dalam *file* ini, perlu dilakukan impor komponen-komponen yang diperlukan dari *React Router*. Berikut adalah komponen-komponen yang wajib diimpor.

c. *Wrap Komponen dengan <Routes>*



```
function App() {  
  return (  
    <div>  
      <Navbar/>  
      <BrowserRouter>  
        <Routes>  
          <Route path="/" element=<Home />/>  
          <Route path="/home" element=<Home />/>  
          <Route path="/tentangkami" element=<TentangKami />/>  
          <Route path="/layanankami" element=<LayananKami />/>  
          <Route path="/kontak" element=<Kontak />/>  
          <Route path="/submissionconfirmed" element=<SubmissionConfirmed />/>  
          <Route path="/companyprofile" element=<CompanyProfile />/>  
          <Route path="/logistics" element=<DetailLogistics />/>  
          <Route path="/sewa" element=<DetailSewa />/>  
          <Route path="/sewajam" element=<SewaJam />/>  
        </Routes>  
      </BrowserRouter>  
    </div>  
  );  
}
```


Gambar 3.7 *Wrap* Komponen `<Routes>`

Untuk menyediakan konteks *routing* maka komponen-komponen harus dibungkus atau *wrap* dengan `<Routes>`

d. Definiskan Komponen untuk Setiap *Route*

```
export default function home(){  
  return (  

```

```
export default function TentangKami(){  
  return(  

```

```
export default function LayananKami(){  
  return(  

```

```
export default function Kontak(){  
  return(  

```

```
export default function SubmissionConfirmed(){  
  return(  

```

Gambar 3.8 Definisi komponen untuk setiap *route*

Menentukan komponen yang akan ditampilkan ketika URL cocok dengan path yang didefinisikan.

e. Navigasi antar *Route*

Untuk membuat navigasi antar *route* maka dapat menggunakan komponen `<a>`. komponen tersebut adalah

komponen dari *React Router* untuk membuat navigasi antar *route* di dalam komponen-komponen.

```
<li className="nav-item">
  <a class="nav-link active" aria-current="page" href="/home"><b>Beranda</b></a>
</li>
<li className="nav-item">
  <div className="batas"></div>
</li>
<li className="nav-item">
  <a class="nav-link" href="/tentangkami"><b>Tentang Kami</b></a>
</li>
<li className="nav-item">
  <div className="batas"></div>
</li>
<li className="nav-item">
  <a class="nav-link" href="/layanankami"><b>Layanan Kami</b></a>
</li>
<li className="nav-item">
  <div className="batas"></div>
</li>
<li className="nav-item">
  <a class="nav-link" href="/kontak"><b>Contact Us</b></a>
</li>
<li className="nav-item">
  <div className="batas"></div>
</li>
<li className="nav-item">
  <a class="nav-link" href="/submissionconfirmed"><b>Submission Confirmed</b></a>
</li>
```

Gambar 3.9 Navigasi antar *Route*

Dengan pengaturan di atas, ketika pengguna menavigasi ke URL yang sesuai (misalnya “tentangkami”), komponen “TentangKami” akan ditampilkan dalam konten aplikasi.

Penggunaan *React Router* memungkinkan pekerja magang membuat aplikasi *React* yang dinamis dengan navigasi yang responsif dan memiliki struktur yang terorganisir.

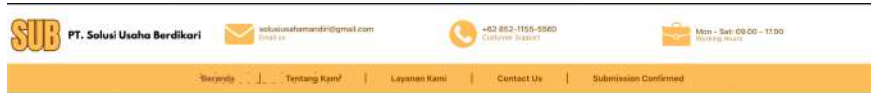
Selain membuat *route*, di minggu ini pekerja magang juga membuat *Homepage website*. *Homepage* atau halaman utama adalah halaman pertama yang ditampilkan ketika pengguna mengunjungi sebuah situs web. halaman ini merupakan titik awal dari navigasi di situs *web* yang dibuat. Tujuan utama dari sebuah *homepage* adalah untuk memberikan informasi yang relevan dan menarik pengunjung agar tetap berada pada situs web dan menjelajahi situs web lebih lanjut.



Gambar 3.10 *HomePage*

Pada tahap ini, pekerja magang merancang homepage yang memiliki tujuan untuk memberikan gambaran umum seputar perusahaan, seperti profil dan apa saja yang ditawarkan perusahaan kepada konsumen. Selain itu pekerja magang juga membuat

Dalam pembuatan homepage ini, pekerja magang membuat struktur kodingan seperti pada gambar 3.11



Gambar 3.12 *Navigation Bar*

Setelah membuat homepage, pekerja magang melanjutkan pekerjaan dengan membuat navigation bar atau navbar dan juga membuat footer, yang nantinya navbar dan footer akan terus muncul pada setiap halaman website. navigation bar menyediakan menu navigasi yang konsisten untuk memudahkan pengguna dalam menjelajahi halaman-halaman atau fitur-fitur yang tersedia pada situs web. navigation bar yang dibuat dapat dilihat pada gambar 3.12



```

1 import React from 'react';
2 import logo from './assets/logo.png';
3 import SUB from './assets/logoSub.png';
4 import MAIL from './assets/mail.png';
5 import CONTACT from './assets/contact.png';
6 import WORK from './assets/work.png';
7
8 export default function Navbar() {
9   return (
10    <div class="header">
11      <div class="logo">
12        <img alt="Logo" src={logo} alt="Logo" style={{ width: '100px'}}/>
13      </div>
14      <div class="sub-header">
15        <div class="mail">
16          <img alt="Mail" src={MAIL} alt="Mail" style={{ width: '100px'}}/>
17        </div>
18        <div class="contact">
19          <img alt="Contact" src={CONTACT} alt="Contact" style={{ width: '100px'}}/>
20        </div>
21        <div class="work">
22          <img alt="Work" src={WORK} alt="Work" style={{ width: '100px'}}/>
23        </div>
24      </div>
25      <div class="contact">
26        <div class="mail">
27          <img alt="Mail" src={MAIL} alt="Mail" style={{ width: '100px'}}/>
28        </div>
29        <div class="contact">
30          <img alt="Contact" src={CONTACT} alt="Contact" style={{ width: '100px'}}/>
31        </div>
32        <div class="work">
33          <img alt="Work" src={WORK} alt="Work" style={{ width: '100px'}}/>
34        </div>
35      </div>
36    </div>
37  );
38}

```

Gambar 3.13 Struktur Kodingan *Navigation Bar*

Struktur Kodingan yang dibuat oleh pekerja magang untuk membuat navigation bar diambil dari template yang sudah disediakan oleh Bootstrap. Pekerja magang mengubah beberapa kodingan untuk menyesuaikan dengan desain yang sudah dibuat, kemudian pekerja magang membuat sebuah header yang berada di atas navigation bar. Struktur kodingan dari navigation bar dapat dilihat pada gambar 3.13



Gambar 3.14 *Footer*

Selain membuat navigation bar, pekerja magang juga membuat footer pada website. Footer sendiri adalah bagian bawah dari sebuah halaman website yang berisi informasi tambahan atau elemen navigasi yang tidak terlalu utama namun tetap penting bagi pengguna. pekerja magang membuat footer pada bagian paling bawah halaman web, dan elemen yang terdapat pada footer yang dibuat tautan navigasi tambahan, informasi alamat, logo perusahaan, serta tautan tautan ke media sosial perusahaan. hasil dari pembuatan footer dapat dilihat pada gambar 3.14

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

```

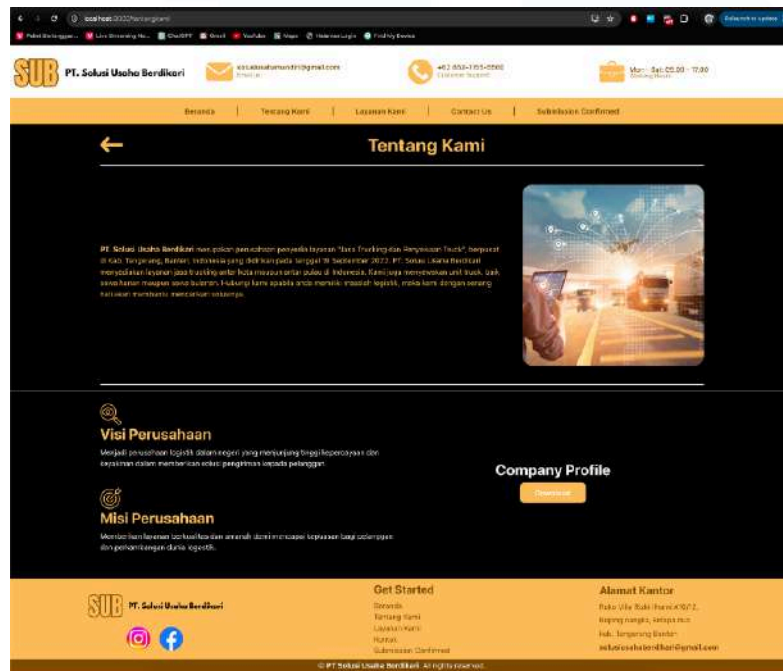
6   export default function Footer(){
7     return (
8       <footer className="footer">
9         <div className="container-fot">
10          <div className="footer-logo">
11            <div className="logo" onClick={() => window.location.href = '/home'} >
12              <img src={SUB} alt="Logo" style={{width: '300px'}}/>
13            </div>
14            <div class="logososmed">
15              <div className="instagram" onClick={() => window.location.href = '/' } >
16                <img src={IG} alt="Logo" style={{width: '50px'}}/>
17              </div>
18              <div className="facebook" onClick={() => window.location.href = '/' } >
19                <img src={FB} alt="Logo" style={{width: '50px'}}/>
20              </div>
21            </div>
22          </div>
23          <div className="footer-nav">
24            <h4><b>Get Started</b></h4>
25            <a class="nav-link" href="/home">Beranda</a>
26            <a class="nav-link" href="/tentangkami">Tentang Kami</a>
27            <a class="nav-link" href="/layanankami">Layanan Kami</a>
28            <a class="nav-link" href="/kontak">Kontak</a>
29            <a class="nav-link" href="/submissionconfirmed">Submission Confirmed</a>
30          </div>
31          <div className="footer-info">
32            <h4><b>Alamat Kantor</b></h4>
33            <p>Ruko Villa Rizki Ilhami A10/12,</p>
34            <p>Bejong nangka, kelapa dua</p>
35            <p>kab. Tangerang Banten</p>
36            <p><b>solusiusahaberdikari@gmail.com</b></p>
37          </div>
38        </div>
39        <div className="footer-copyright">
40          <div className="copyright-content">
41            <p>&copy; <b>PT Solusi Usaha Berdikari</b>. All rights reserved.</p>
42          </div>
43        </div>
44      </footer>

```

Gambar 3.15 Struktur Kodingan *Footer*

Pada gambar diatas dapat dilihat struktur kodingan yang dibuat oleh pekerja magang untuk merancang footer. Komponen footer sendiri dibungkus atau di warp dengan `</footer>`. kemudian untuk isi dari footer sendiri diisi dengan beberapa informasi seperti yang dapat dilihat pada gambar 3.15

3.2.4 Membuat Page Tentang Kami



Gambar 3.16 Page Tentang Kami

Pada minggu ketiga, pekerja magang membuat page tentang kami, page ini berisikan profil dari perusahaan dan penjelasan singkat mengenai perusahaan. Selain profil perusahaan, page ini juga berisikan visi dan misi perusahaan, dan pekerja magang juga menambahkan company profile yang dapat dilihat dan didownload oleh pengguna. visual dari page tentang kami dapat dilihat pada gambar 3.16

```

import React from "react";
import companyprofile from '../assets/CompanyProfile.pdf'

export default function CompanyProfile(){
  return(
    <embed src={companyprofile} width="100%" height="1200"></embed>
  );
}

```

Gambar 3.17 Warp <embed>

Pada *page* tentang kami, terdapat *company profile* yang dapat dibuka dan diunggah oleh pengguna dalam format pdf. Pekerja magang membuat *file* pdf yang kemudian diupload ke website dengan menggunakan warp <embed> file *company profile* adalah file yang sudah disimpan pada *folder assets* yang kemudian di import ke dalam kodingan dengan menggunakan elemen import dari *React*. dokumentasi dari ^ <embed> dapat dilihat pada gambar 3.16

3.2.5 Membuat *Page Layanan Kami*

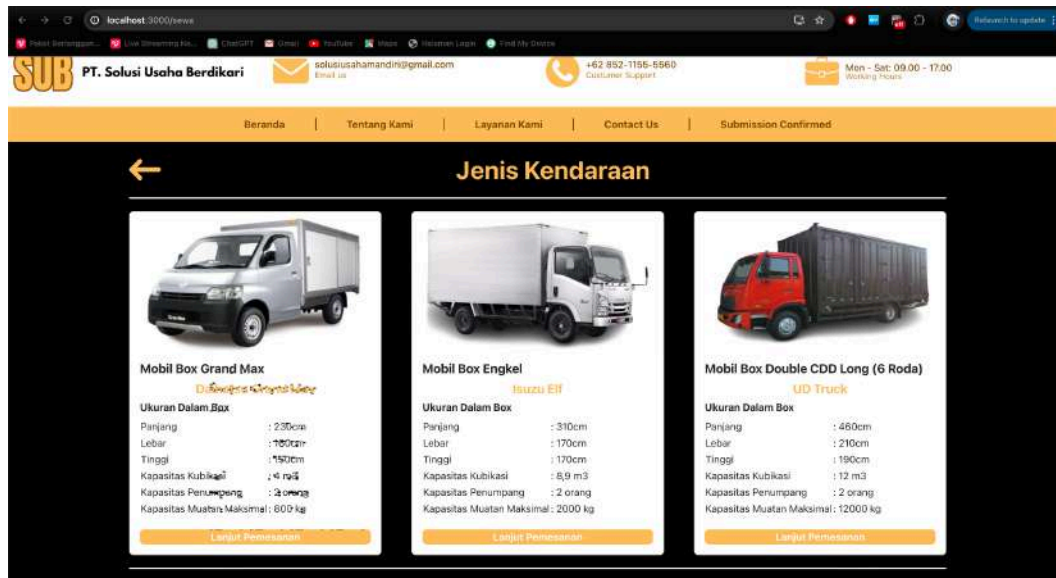
Pekerja magang juga membuat *page* layanan kami yang berisikan dua layanan, yaitu layanan logistik dan layanan penyewaan *truck*. Pada *page* tersebut terdapat tombol “selengkapnya”, tombol tersebut merupakan fitur yang mengarahkan pengguna kepada detail dari layanan yang ditawarkan oleh perusahaan.



Gambar 3.17 Page Layanan Kami

Pada page layanan logistik, belum terlampir detail-detail dari layanan tersebut. Hal ini dikarenakan harga dan layanan tersebut belum dipastikan lagi oleh perusahaan. Untuk *age* tersebut akan diselesaikan oleh pekerja magang apabila sudah mendapatkan briefing lebih lanjut dari perusahaan.

Pada page layanan penyewaan *truck*, pekerja magang membuat detail layanan dengan menggunakan komponen “*card*”. Komponen “*card*” adalah salah satu komponen yang sangat umum dan berguna untuk membangun tata letak agar visual yang ditampilkan menjadi lebih terstruktur dan menarik. Komponen “*card*” juga digunakan untuk menampilkan konten dalam sebuah kotak yang dapat diatur dengan berbagai elemen seperti gambar, teks, dan juga tombol. Pada *page* ini konten yang disimpan dalam komponen “*card*” berupa gambar, teks, dan tombol yang pekerja magang rancang apabila pengguna menekan tombol tersebut akan mengarahkan (*direct*) pengguna pada aplikasi “WhatsApp”, karena tombol tersebut adalah tombol untuk melakukan pemesanan unit *truck*. Dokumentasi dari page layanan penyewaan truck dapat dilihat pada gambar 3.18.



Gambar 3.18 Page Layanan Penyewaan Truck

Page penyewaan truck diisi dengan jenis kendaraan yang disewakan dan juga layanan dari penyewaan truck. Pekerja magang menampilkan jenis layanan tersebut dalam *website* juga menggunakan komponen “*card*”. Pada bagian ini tombol yang terdapat dalam “*card*” merupakan tombol yang mengarahkan pengguna untuk melihat detail harga yang ditawarkan oleh perusahaan. Dokumentasi dari jenis layanan dapat dilihat pada gambar 3.19.



Gambar 3.19 Jenis Layanan

Struktur kodingan pada komponen “*card*” adalah struktur yang sudah disediakan oleh Bootstrap, kemudian pekerja magang melakukan modifikasi kodingan tersebut untuk menyesuaikan dengan kebutuhan dan menyesuaikan dengan konten yang akan diisi pada *card*. Struktur kodingan yang dibuat oleh pekerja magang dapat dilihat pada gambar 3.20.

UMN
UNIVERSITAS
MULTIMEDIA
NUSANTARA

```

20 <div class="card" style={{width: '400px'}}>
21 <img src={GrandMax} class="card-img-top" alt="grandmax"></img>
22 <div class="card-body">
23 <h5 class="card-title"><b>Mobil Box Grand Max</b></h5>
24 <h5 class="tit"><b>Daihatsu Grand Max</b></h5>
25 <h6><b>Ukuran Dalam Box</b></h6>
26 <table>
27 <tr>
28 <td>Panjang</td>
29 <td>: 230cm</td>
30 </tr>
31 <tr>
32 <td>Lebar</td>
33 <td>: 160cm</td>
34 </tr>
35 <tr>
36 <td>Tinggi</td>
37 <td>: 150cm</td>
38 </tr>
39 <tr>
40 <td>Kapasitas Kubikasi</td>
41 <td>: 4 m3</td>
42 </tr>
43 <tr>
44 <td>Kapasitas Penumpang</td>
45 <td>: 2 orang</td>
46 </tr>
47 <tr>
48 <td>Kapasitas Muatan Maksimal</td>
49 <td>: 800 kg</td>
50 </tr>
51 </table>
52 <a href="https://wa.me/6285211555560" class="bton"><b>Lanjut Pemesanan</b></a>
53 </div>
54 </div>

```

Gambar 3.20 Struktur Kodingan Card

Pada bagian detail jenis layanan, pekerja magang membuatnya dengan menggunakan tabel. pembuatan tabel sendiri menggunakan tag `<table>`, `<th>` `<td>`, dan `<tr>`. Tag `<table>` digunakan untuk untuk mendefinisikan sebuah tabel dalam HTML. Dokumentasi dari pembuatan tabel dapat dilihat pada gambar 3.21

```

<table class="tabelharga">
  <tr>
    <th>JENIS KENDARAAN</th>
    <th>HARGA (JABODETABEK)</th>
    <th>OVERTIME</th>
  </tr>
  <tr>
    <td>GRAND MAX</td>
    <td>4.000.000,-</td>
    <td>500.000,-</td>
  </tr>
  <tr>
    <td>ENGKEL 4 RODA</td>
    <td>11.500.000,-</td>
    <td>900.000,-</td>
  </tr>
  <tr>
    <td>DOUBLE 6 RODA / CDD LONG</td>
    <td>12.500.000,-</td>
    <td>1.300.000,-</td>
  </tr>
</table>

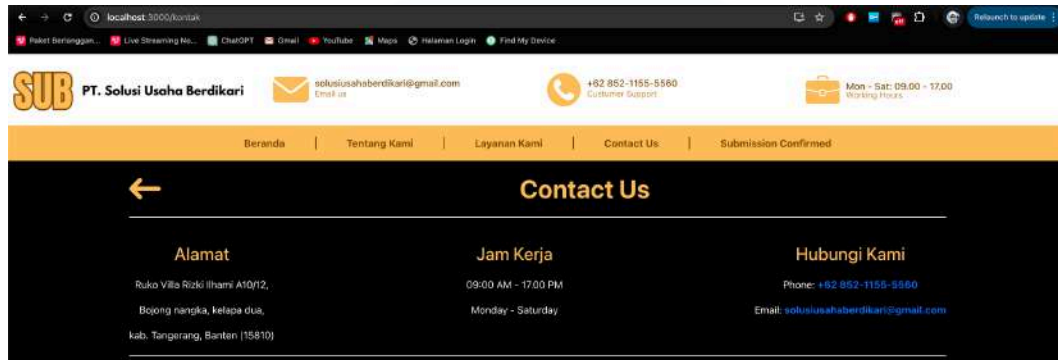
```

Gambar 3.21 Struktur Kodingan Pembuatan Tabel

3.2.6 Membuat *Page Contact Us*

Page contact us adalah *page* yang berisikan alamat kantor, jam kerja kantor dan juga kontak berupa nomor telepon dan whatsapp serta email yang

dapat dihubungi konsumen untuk menghubungi perusahaan. Dokumentasi dari page contact us dapat dilihat pada gambar 3.22



Gambar 3.22 Page Contact Us

Untuk nomor telepon dan email, pekerja magang membuatnya dengan elemen `<a>`. Elemen tersebut adalah elemen dalam HTML yang digunakan untuk membuat tautan (*link*) ke halaman *web* lain. pekerja magang menambahkan nomor telepon dan no whatsapp dengan menggunakan elemen `<a>` untuk memudahkan konsumen apabila ingin menghubungi pihak perusahaan, karena apabila konsumen menekan nomor yang tertera pada *website*, maka konsumen akan diarahkan pada whatsapp *web*. begitu juga dengan email, apabila konsumen menekan email yang tertera pada *website* maka konsumen akan langsung diarahkan pada aplikasi email yang terdapat pada perangkat konsumen. Struktur kodingan dari *page contact us* dapat dilihat pada gambar 3.23.


```
import React from "react";
import BACK from '../assets/back2.png';
//import Navbar from '../component/Navbar';

export default function Kontak(){
  return(
    <div class="container">
      <div class="head">
        <div className="back" onClick={() => window.location.href = '/home'} >
          <img src={BACK} alt="but" style={{width: '50px'}}></img>
        </div>
        <h1 Contact Us</h1>
      </div>
      <div class="garis"></div>
      <div class="contactus">
        <div class="alamat">
          <h3 class="text">Alamat</h3>
          <p>Ruko Villa Rizki Ilhami A10/12,</p>
          <p>Bojong nangka, kelapa dua,</p>
          <p>kab. Tangerang, Banten (15810)</p>
        </div>
        <div class="Jam Kerja">
          <h3 class="text">Jam Kerja</h3>
          <p>09:00 AM - 17.00 PM</p>
          <p>Monday - Saturday</p>
        </div>
        <div class="kontak">
          <h3 class="text">Hubungi Kami</h3>
          <p>Phone: <b><a href="https://wa.me/6285211555560">+62 852-1155-5560</a></b></p>
          <p>Email: <b><a href="mailto:solusiusahamendir@gmail.com">solusiusahaberdikari@gmail.com</a></b></p>
        </div>
      </div>
      <div class="garis"></div>
    </div>
  )
}
```

Gambar 3.23 Struktur Kodingan *Page Contact Us*

3.3.7 Membuat *Map Direction*

Pada Tahap ini, pekerja magang pertama-tama membuat titik koordinat kantor dengan menggunakan fitur yang tersedia pada *Google Map*. setelah titik Koordinat dibuat kemudian pekerja magang menyalin HTML code yang diberikan *google map*. Dokumentasi HTML *code* dari *google maps* dapat dilihat pada gambar 3.24

Bagikan



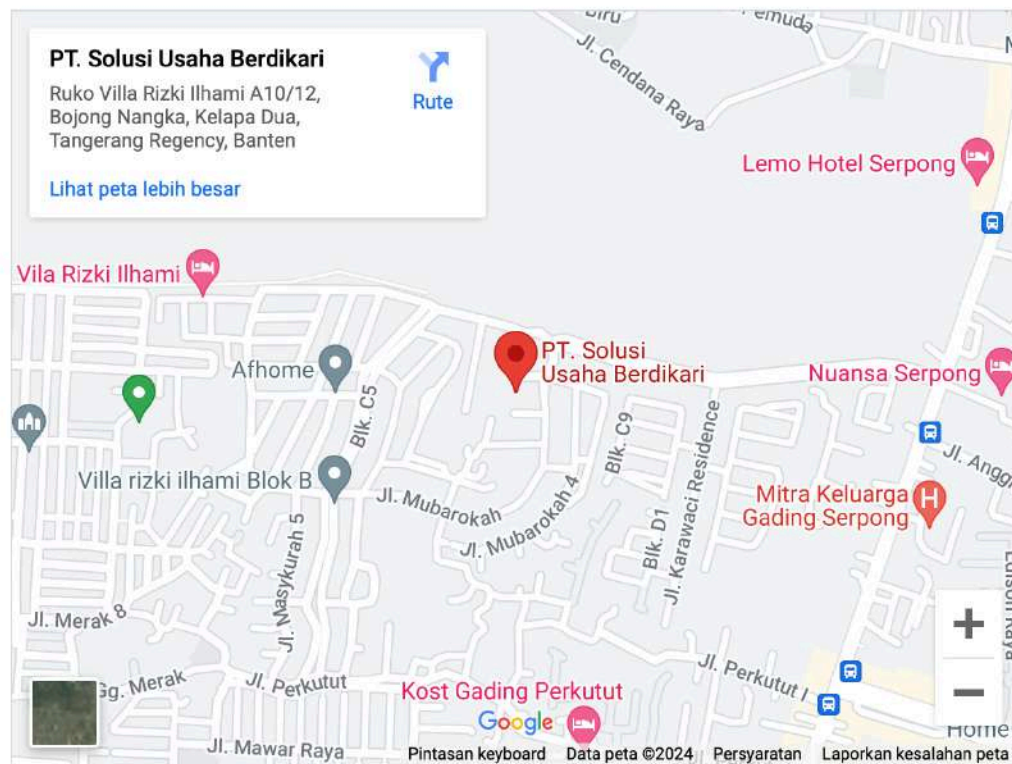
Kirim link

Sematkan peta

Sedang ▾

<iframe src="https://www.google.com/maps/embed?pb=!1m18!1m12!"

SALIN HTML



Dengan menyematkan peta ini, Anda setuju dengan [Persyaratan Layanan](#).

Gambar 3.24 HTML code dari Google Maps

Setelah code disalin, kemudian pekerja magang memasukkan *code* tersebut ke dalam kodingan dari proyek yang sedang dikerjakan dengan menggunakan elemen **<iframe>**. Elemen **<iframe>** dalam HTML adalah elemen yang digunakan untuk menyematkan dokumen HTML atau konten dari sumber eksternal ke dalam dokumen HTML yang sedang dikerjakan oleh pekerja magang. Elemen ini juga memungkinkan untuk menampilkan konten dari satu halaman *web* di dalam area terpisah di dalam halaman *web* lain. Selain untuk menyematkan *maps*, elemen **<iframe>** juga dapat menyematkan video, gambar,

dan lain sebagainya. Struktur kodingan yang dibuat pekerja magang untuk menyematkan map direction dapat dilihat pada gambar 3.25.

```
<div className="google-map">
  <iframe
    title="Google Maps"
    src="https://www.google.com/maps/embed?pb=!1m18!1m12!1m3!1d317,986531162152412d106,6045485581835113d-6,24790000715586312m3!1f0!2f0!3m2!111024121768!4f1
    width="600"
    height="450"
    style={{ border: 0 }}
    allowFullScreen=""
    loading="lazy"
    referrerPolicy="no-referrer"
  ></iframe>
```

Gambar 3.25 Struktur Kodingan *Map Direction*

3.3.8 Membuat *Website Responsive*

Pada tahap ini, pekerja magang membuat *website* menjadi *responsive*. *Website responsive* merupakan *website* yang memiliki orientasi konten yang dinamis menyesuaikan ukuran pada layar pengguna *website*. Untuk membuat *website* yang *responsive*, pekerja magang menggunakan *framework Bootstrap*. Pekerja magang menggunakan *framework* tersebut karena *Bootstrap* menyediakan komponen-komponen yang *responsive* dan mudah diimplementasikan dalam proyek *React*.

untuk mengimplementasikan *Bootstrap* pada proyek *React*, pekerja magang memasukan link yang sudah disediakan pada *website Bootstrap*, link tersebut kemudian di *copy-paste* di bagian **<head>** Pada file **index.html**. untuk dokumentasi implementasi *framework bootstrap* pada *react* dapat dilihat pada gambar 3.26.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 | <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU90FePok6YctnYm05pKlyT2hRjXh03"
5 <meta charset="utf-8" />
6 <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
7 <meta name="viewport" content="width=device-width, initial-scale=1" />
8 <meta name="theme-color" content="#000000" />
9 <meta
10 name="description"
11 content="Web site created using create-react-app"
12 />
13 <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
14 <!--
15 manifest.json provides metadata used when your web app is installed on a
16 user's mobile device or desktop. See https://developers.google.com/web/fundamentals/web-app-manifest/
17 -->
18 <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
19 <!--
20 Notice the use of %PUBLIC_URL% in the tags above.
21 It will be replaced with the URL of the 'public' folder during the build.
22 Only files inside the 'public' folder can be referenced from the HTML.
23
24 Unlike "/favicon.ico" or "favicon.ico", "%PUBLIC_URL%/favicon.ico" will
25 work correctly both with client-side routing and a non-root public URL.
26 Learn how to configure a non-root public URL by running npm run build.
27 -->
28 <title>React App</title>
29 </head>
```

Gambar 3.26 Implementasi *Framework Bootstrap* pada *React*

Selain dengan memasukan link yang sudah disediakan *Bootstrap* pada proyek *React* yang sudah disediakan, terdapat cara lain yang dapat diimplementasikan untuk menginstall *Bootstrap* pada proyek *React*, yaitu dengan menginstall langsung pada bash atau terminal dari proyek *React*. Untuk menginstall *Framework Bootstrap* melalui *bash* atau terminal, perlu dilampirkan **npm install bootstrap** pada *bash* atau terminal kemudian melakukan import pada proyek yang sedang dikerjakan dengan format **import 'bootstrap/dist/css/bootstrap.min.css'**; dokumentasi nya dapat dilihat pada gambar 3.27

```
○ (base) christoperjohn@192 app % npm install bootstrap
```

Gambar 3.27 *Install Bootstrap* melalui *bash* atau terminal

3.3.9 *Hosting Website*

Pada tahap ini, pekerja magang melakukan *hosting* untuk proyek *website* yang sedang dikerjakan. *Hosting* sendiri adalah proses menyediakan dan menjalankan situs *website*, atau layanan *online* di server yang dapat diakses

melalui internet. *Hosting* sangat penting karena dengan *hosting* maka konten dari *website* perusahaan dapat diakses oleh pengguna melalui browser.

Pekerja magang melakukan *hosting website company profile* yang dibangun pada *website* “**my.idcloudhost.com**”. *Website* tersebut adalah *website* yang memberikan layanan *Web Hosting*, *Domain Registration*, *Cloud Hosting*, *VPS (Virtual Private Server) Hosting*, *PaaS (Platform as a Service) Control Panel*, dan sebagainya. Tahap awal yang dilakukan pekerja magang untuk *hosting website company profile* adalah melakukan registrasi domain, dimana PT Solusi Usaha Berdikari meminta agar *website company profile* yang sedang dibangun di *hosting* dengan domain “solusiusahaberdikari.com”. Untuk memenuhi permintaan tersebut, pekerja magang melakukan registrasi domain sesuai dengan permintaan dari perusahaan. Dokumentasinya dapat dilihat dari gambar 3.28



Gambar 3.28 Domain yang Digunakan untuk *Hosting*

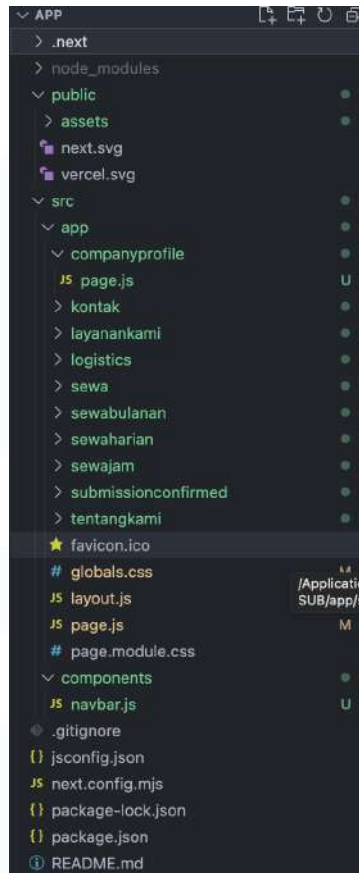
Service cloud yang digunakan adalah *shared hosting* yang disediakan oleh **IdCloudHost**. *shared hosting* merupakan layanan *hosting* dimana situs website berbagi sumber daya *server* yang sama. *deployment* dilakukan pada *control panel* yang memiliki spesifikasi *Disk Space* 3 GB, *CPU* 1 Core, *RAM* 1 GB, *Subdomain Unlimited*, *Bandwidth Unlimited*, *Database Unlimited*. Spesifikasi tersebut unggul dari segi fleksibilitas, meskipun sederhana hal tersebut cukup untuk banyak proyek dan memungkinkan pengelolaan yang cukup besar untuk aplikasi dan situs web yang sedang berkembang.

Tahap awal melakukan *deployment* aplikasi Next.js yaitu dengan melakukan build pada project yang sedang dikerjakan. *Build* dilakukan dengan memasukan perintah “**npm run build**” pada terminal, setelah project selesai di build, maka tahap selanjutnya adalah melakukan *compress* pada folder project yang sedang dikerjakan. Setelah itu folder yang di *compress* dimasukkan ke dalam

cpanel dan kemudian di ekstrak di *cpanel*. Setelah folder di ekstrak, tahap selanjutnya adalah melakukan *set-up Node.js App*, *set-up* yang dilakukan yaitu menentukan versi dari *Node.js*, menentukan *application mode*, menentukan *application URL*, dan menentukan *application startup file*. setelah itu tekan tombol “*create*” dan proses *deployment* selesai dilakukan.

3.3.10 Migrasi *Website* CSR menjadi SSR

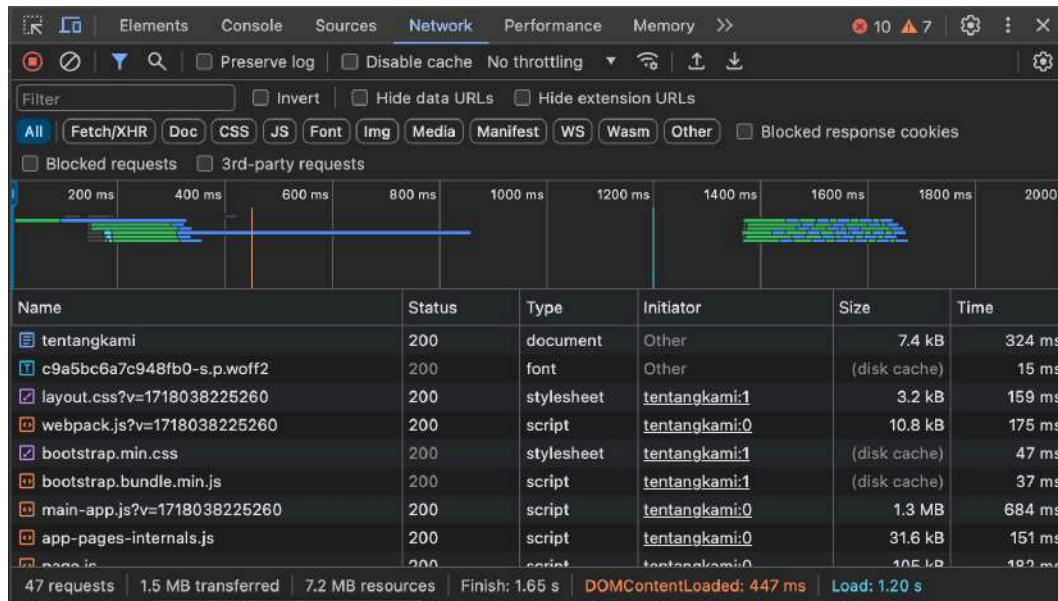
Pada tahap ini, pekerja magang melakukan migrasi pada *website* yang dikerjakan dari CSR (*Client-Side-Rendering*) menjadi SSR (*Server-Side-Rendering*). Alasan migrasi dilakukan adalah agar SEO (*Search Engine Optimization*) lebih baik, Hal tersebut dikarenakan *Search engine bots* lebih mudah mengindeks konten yang telah di *render* di server. selain itu, *website* SSR unggul dalam keamanan dan waktu interaktif yang lebih cepat karena halaman sudah di *render* sebelum mencapai *browser*. Proses migrasi dilakukan dengan menggunakan Next.js. Dokumentasi *set-up* menggunakan Next.js dapat dilihat pada gambar3.29.



Gambar 3.29 Set-up Next.js

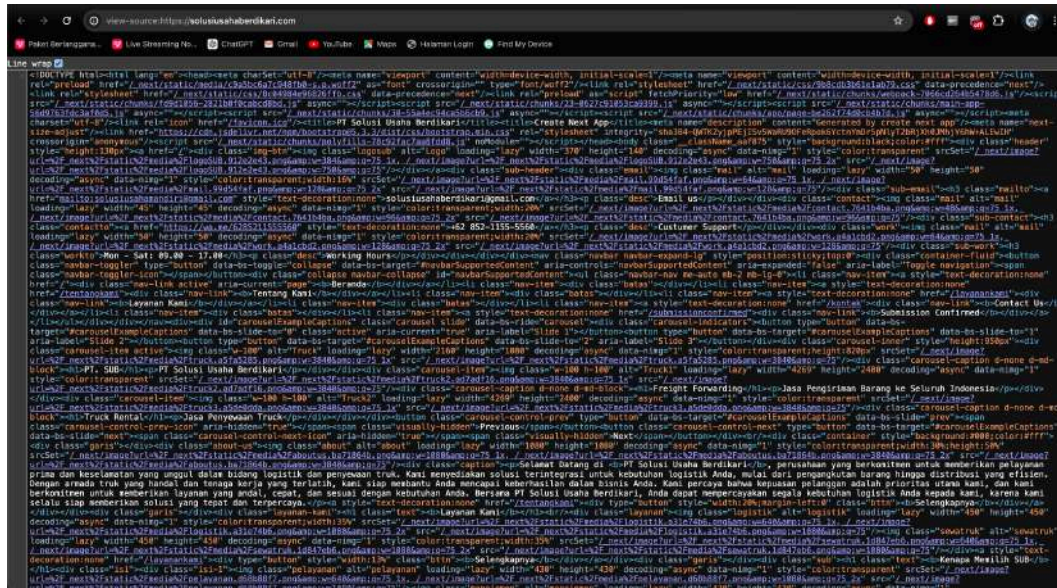
proses-proses tersebut meliputi *set-up* Next.js, migrasi *file*, *routing* dan *testing*. Dokumentasi dari migrasi website CSR menjadi SSR dapat dilihat pada gambar 3.30.

UIN
UNIVERSITAS
MULTIMEDIA
NUSANTARA



Gambar 3.30 Website yang sudah SSR

SSR (*Server-Side-Rendering*) adalah adalah teknik *rendering* halaman *website* dimana server merender halaman lengkap (HTML) dan mengirimkannya ke *client* (*browser*) sebelum halaman tersebut ditampilkan. Untuk mengetahui bahwa website SSR adalah dengan cara membuka “*view page source*” pada halaman *website*. Apabila elemen-elemen yang terdapat pada halaman muncul di *source*, maka dapat dipastikan website tersebut adalah *website* SSR karena semuanya di *render* oleh *server*. Perbedaan dengan website CSR adalah apabila membuka “*view page source*” maka hasilnya *blank* atau tidak ada elemen-elemen yang tampil di *sourcenya*. Dokumentasi dari halaman *website* SSR dapat dilihat pada gambar 3.31.



Gambar 3.31 Page source (HTML yang bisa diindex browser)

3.3 Kendala yang Ditemukan

Selama melaksanakan praktek kerja magang pada PT Solusi Usaha Berdikari, pekerja magang menemukan dan merasakan beberapa kendala. Berikut adalah beberapa kendala yang ditemukan oleh pekerja magang selama melaksanakan praktek kerja magang:

1. Kurangnya informasi yang jelas pada saat memberikan tugas. Hal ini membuat pekerja magang kesulitan dalam mengerjakan tugas tersebut.
2. Perusahaan kerap memberikan tugas kepada pekerja magang pada hari libur. Hal ini menghambat pekerja magang dalam pembuatan laporan kerja magang, karena pekerja magang menggunakan hari libur untuk mengerjakan laporan kerja magang.

3.4 Solusi atas Kendala yang Ditemukan

Dengan adanya kendala yang ditemukan dan dirasakan, tentu saja terdapat solusi yang ditemukan oleh pekerja magang dalam mengatasi kendala tersebut. Berikut adalah penjabaran mengenai solusi untuk menjawab kendala diatas.

1. Memberikan informasi yang jelas pada saat memberikan tugas atau pekerja magang lebih aktif bertanya apabila belum memahami tugas yang diberikan.
2. Perusahaan tidak memberikan tugas kepada pekerja magang pada hari libur, melainkan memberikan tugas pada jam kerja.

