

BAB III PELAKSANAAN KERJA PROFESI

3.1 Bidang Kerja

IT Development di PT ASLI Rancangan Indonesia bertanggung jawab dalam menangani sistem yang sudah berjalan dengan baik, serta memastikan semua sesuai dengan tujuan perusahaan. Selama menempuh kerja profesi di PT ASLI Rancangan Indonesia, penulis dipercaya oleh Project & Product Manager untuk membantu divisi *Information Technology (IT)* sebagai *Full-Stack Developer*. *Full-Stack Developer* adalah sebuah posisi dimana seorang programmer bekerja dengan teknologi *Back-end* dan *Front-end* secara bersamaan. Tugas ini meliputi pembuatan basis data (*database*), penggunaan *JavaScript*, *Next.js*, *integrasi API*, serta penyesuaian kebutuhan antarmuka pengguna (UI) pada website.

Penulis mendapatkan tugas untuk melakukan *reframing* dan *reengineering* terhadap website PT Bank Abdi. Tugas ini diberikan karena sebelumnya website PT Bank Abdi bersifat statis dan tidak memiliki fitur yang memungkinkan untuk memperbarui konten maupun isi halaman secara dinamis. Setiap kali diperlukan perubahan atau penambahan konten, administrator atau pihak terkait harus menghubungi pengembang. Hal ini mengharuskan pengembang melakukan pembaruan langsung pada *source code* agar isi konten dan halaman website dapat diperbarui.

Penulis sebagai developer dengan posisi *Full-Stack Developer*, diberikan kesempatan untuk menambahkan beberapa fitur tambahan pada halaman dashboard admin seperti, penambahan fitur *tracking visitor* pada website PT Bank Abdi yang telah di perbarui sistemnya.

Sebelum memulai pekerjaan tersebut, penulis terlebih dahulu mendapatkan penjelasan mengenai ruang lingkup pekerjaan yang akan dilaksanakan serta langkah-langkah yang harus diambil. Hal ini bertujuan agar penulis memahami alur pengerjaan proyek secara menyeluruh, dan dapat menyesuaikan pekerjaan dengan *timeline* serta teknologi (*engine*) yang digunakan oleh perusahaan dan disepakati oleh pihak PT Bank Abdi.

3.2 Pelaksanaan Kerja

Pelaksanaan kerja profesi dimulai dari tanggal 3 Februari 2025 hingga selesai pada 2 Mei 2025. Kerja profesi telah dilakukan selama 90 hari kerja. Jam kerja yang di terapkan perusahaan PT ASLI Rancangan Indonesia dimulai pada pukul 9.30 WIB dan berakhir pada pukul 18.00 WIB. Dalam mengerjakan proyek yang diberikan, penulis membuat website dinamis dengan mengacu pada metode **SDLC (System Development Life Cycle)**. Metode SDLC yang dipakai adalah metode Agile. Dengan metode Agile, pada sistem ini mengacu pada metode **Extreme Programming (XP)**, karena lebih berfokus kepada teknis dan menghasilkan produk yang berkualitas tinggi. XP mencakup perencanaan (*Planning*), Desain, *Coding*, *Testing*, dan *Feedback* dari pengguna. Dalam proyek ini karena desain sama dengan yang sebelumnya maka tahap desain di rubah menjadi tahap *pra-development*

Selama waktu kerja profesi, Berikut adalah rincian pekerjaan yang saya lakukan selama menjalani kerja profesi di perusahaan, berikut tabel uraiannya :

Tabel 3. 1 tabel agenda waktu pelaksanaan

No	Nama Kegiatan	Agenda	Agenda Waktu Pelaksanaan															
			Februari				Maret				April				Mei			
			1	2	3	4	1	2	3	4	1	2	3	4	1			
1	Planning	Perkenalan lingkungan Kerja																
		Penentuan <i>framework</i> yang akan digunakan																
		Belajar mengenai sistem Next.js																
2	Pra-Developme nt	Mencari assets dan <i>library</i> yang akan dipakai																
		Membuat <i>Flowchart</i>																

		Membuat UseCase Diagram																
		Menentukan "Role" admin yang ada untuk sistem dashboard																
3	Development	Scaling UI dari website yang sudah ada																
		Membuat ERD untuk <i>back-end</i>																
		Membuat sistem <i>back-end</i> CMS																
4	Testing	Menguji kelayakan website dan sistem CMS																
5	Feedback	Presentasi kepada Bank-Abdi																

3.2.1 Planning

Perencanaan yang dilakukan di awal mencakup pengenalan lingkungan kerja, *Lead* dari tim *Front-end* dan tim *back-end* memberikan waktu beberapa hari untuk membiasakan diri dengan lingkungan kerja. Mulai dari perkenalan divisi IT yang ada serta berkenalan dengan tim *Front-end* dan *back-end* lainnya. Selanjutnya *lead Front-end* memberikan waktu beberapa hari untuk mempelajari *framework* yang akan dipakai yaitu Next.JS, Prisma(Next.JS) serta repository Github yang akan dipakai.

Pada sistem *Back-end*, penulis di anjurkan untuk memakai database SQL yaitu PostgreSQL untuk lebih gampang dalam menyimpan data, karena di perusahaan lebih dominan menggunakan PostgreSQL untuk databasenya, serta menggunakan Prisma(Next.JS) untuk ORMnya.

3.2.2 Pra-Development

Setelah melakukan perencanaan awal terkait kebutuhan *framework* yang akan digunakan, penulis mulai mengumpulkan berbagai aset gambar dan *library* pendukung lainnya yang akan membantu dalam pengembangan website dinamis. Beberapa di antaranya termasuk *library* untuk animasi seperti *Framer Motion* dan *React-Quill* yang akan digunakan pada antarmuka dashboard untuk penulisan konten baru. Langkah ini dilakukan agar pada tahap pengembangan (*development*), proses implementasi dapat langsung dilakukan sesuai dengan perencanaan awal, seperti pemilihan nama *library*, aset yang digunakan, serta pembaruan aset terbaru yang relevan dalam membangun website yang interaktif dan dinamis.

Selain itu Pada tahap Pra-Development juga ditentukan jumlah peran (*role*) yang akan digunakan untuk sistem dashboard administrator. Selain admin yang bertugas akan ada *role* dengan nama Operator yang bertugas untuk memperbarui isi konten. Perbedaan antara administrator dan operator adalah, administrator sebagai *role* tertinggi dimana admin dapat mengubah, melihat perubahan konten serta dapat menambahkan beberapa operator yang akan bertugas untuk memperbarui isi konten.

Operator hanya dapat melihat, mengubah isi konten namun tidak dapat menambahkan/melihat list *role* operator lainnya. Sebagai contoh di bawah ini merupakan **gambar 3.1** tampilan milik administrator



Gambar 3. 1 Tampilan Dashboard admin

Di tampilan dashboard milik administrator pada **gambar 3.1** dengan nama administrator “reyki” memiliki halaman operator jika dibuka halamannya akan berisi list daftar operator dan juga dapat menambahkan *role* operator.



Gambar 3. 2 Tampilan menu operator

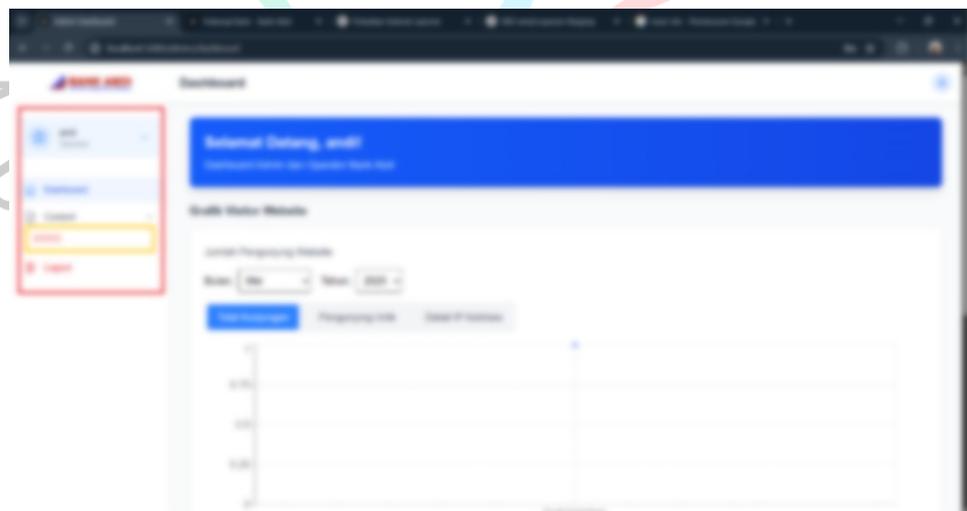
Pada **gambar 3.2** terdapat 2 operator yang bertugas yaitu dengan nama Femas dan juga Andi, jika menjadi administrator dapat melihat jumlah operator yang ada.



Gambar 3. 3 Tampilan tambah Operator

Dan pada gambar diatas, merupakan tampilan menambah operator baru, jadi administrator dapat menambah operator yang bertugas.

Untuk *role* Operator, hanya berbeda pada tampilan UI dimana tidak ada menu operator jika login dengan *role* operator, tampilannya seperti dibawah ini:



Gambar 3. 4 tampilan dashboard jika login dengan role OPERATOR

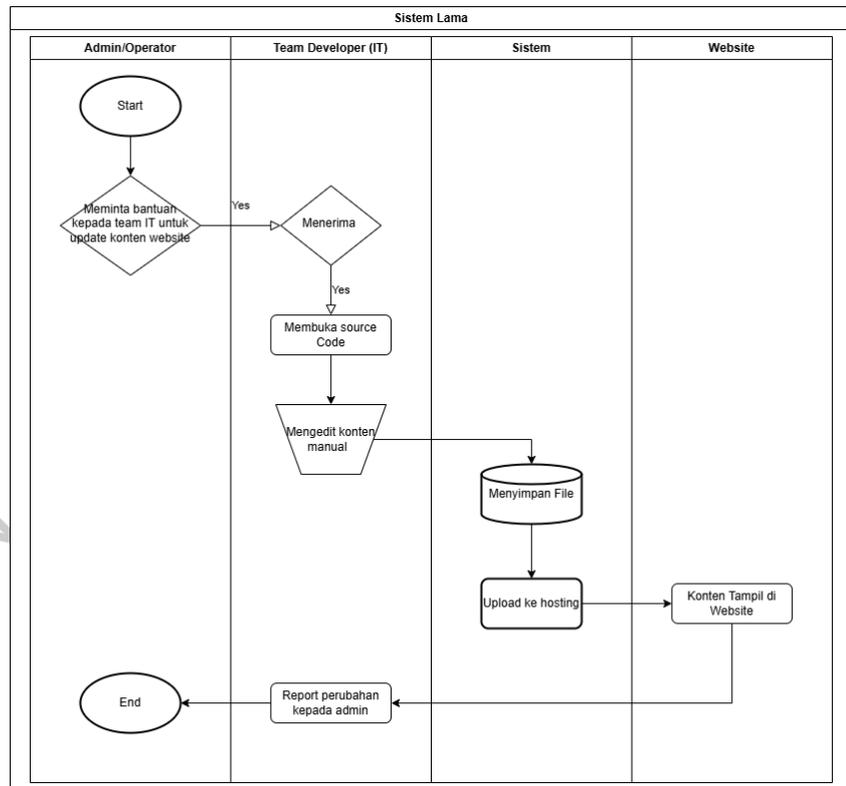
Tidak ada menu “Operator” jika login dengan *role* operator, karena hanya admin yang dapat membuka, melihat, dan menambahkan operator.

Dalam tahap *pra-development*, penulis juga melakukan berbagai persiapan penting untuk menunjang proses pengembangan sistem secara lebih terstruktur dan terarah. Beberapa kebutuhan yang disiapkan meliputi pembuatan *flowchart* program untuk memetakan alur logika sistem, penyusunan *use case diagram* untuk menggambarkan interaksi antara aktor dan sistem, serta perumusan spesifikasi *use case* yang berfungsi untuk menjelaskan secara rinci setiap skenario penggunaan yang terjadi dalam sistem. Seluruh elemen tersebut disusun agar proses *development* dapat berjalan lebih efisien dan sesuai dengan kebutuhan yang telah dianalisis sebelumnya.

1. Membuat Flow Chart Program & Sistem

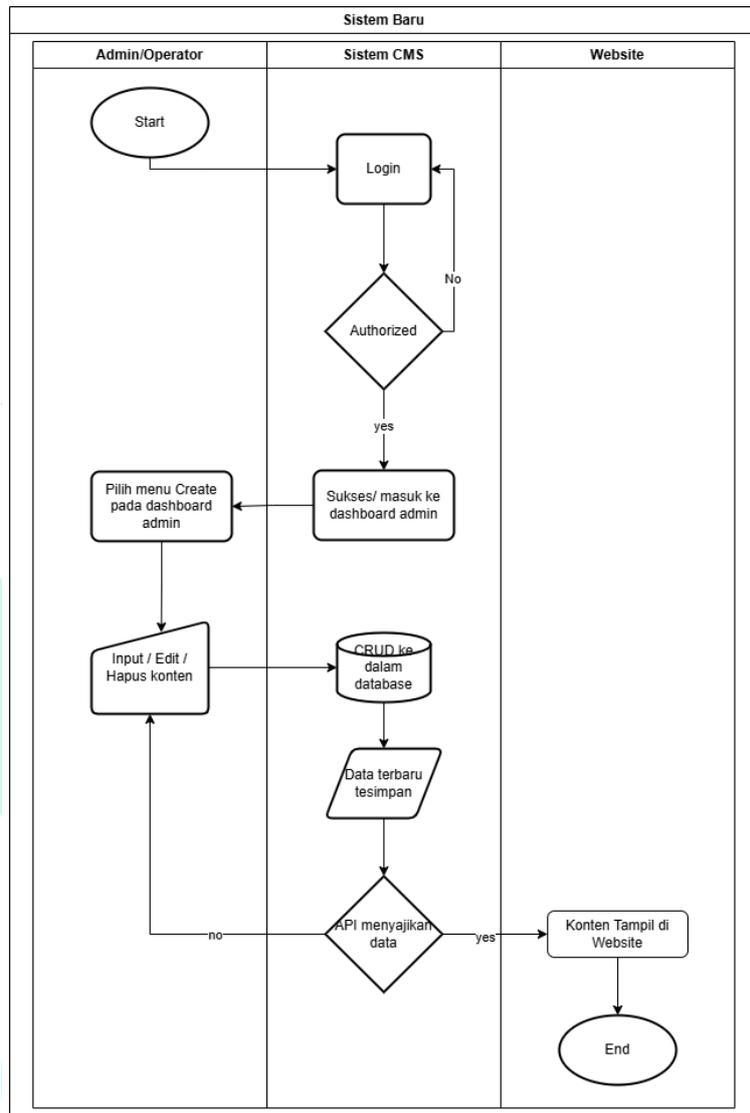
Flowchart program adalah sebuah diagram yang menampilkan langkah-langkah serta keputusan untuk sebuah proses yang dilakukan oleh program. Dalam *flowchart* semua langkah di dokumentasikan dalam bentuk diagram dan dihubungkan dengan garis / panah. Fungsi dari *flowchart* dalam suatu program adalah untuk memberikan gambaran atau alur dari program tersebut dari satu proses ke proses lain agar mudah dipahami oleh semua orang.

Dalam hal ini, penulis akan memberikan beberapa *flowchart* program, *Flowchart* Program sistem lama, *Flowchart* Program sistem baru, *Flowchart* sistem fetchData ke *Front-end*, *Flowchart* hak akses admin dan juga operator. *Flowchart* yang dirancang untuk sistem CMS website PT Bank Abdi digambarkan seperti dibawah ini:



Gambar 3. 5 Flowchart sistem lama

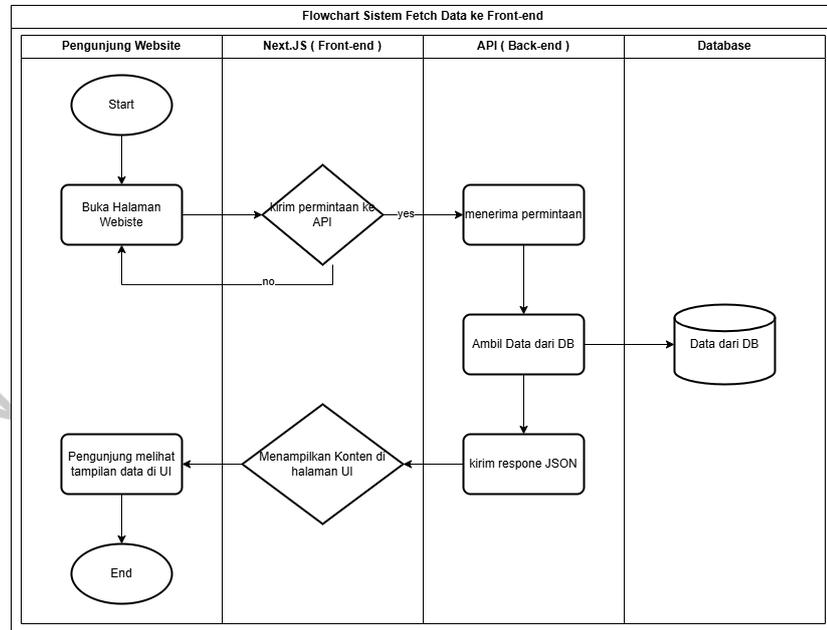
Pada **gambar 3.5**, menampilkan *Flowchart* sistem pada website milik PT Bank Abdi yang sebelumnya. *Flowchart* tersebut menjelaskan bahwa jika admin/operator ingin mengubah isi dari konten halaman pada website harus menghubungi tim *developer* terlebih dahulu, dan tim *developer* harus memperbarui konten secara static dengan cara mengganti langsung *source code* yang ada. Hal ini menyebabkan permasalahan karena setiap perubahan memerlukan waktu yang tidak singkat untuk memperbarui isi konten pada halaman website tersebut.



Gambar 3. 6 Flowchart Sistem Baru

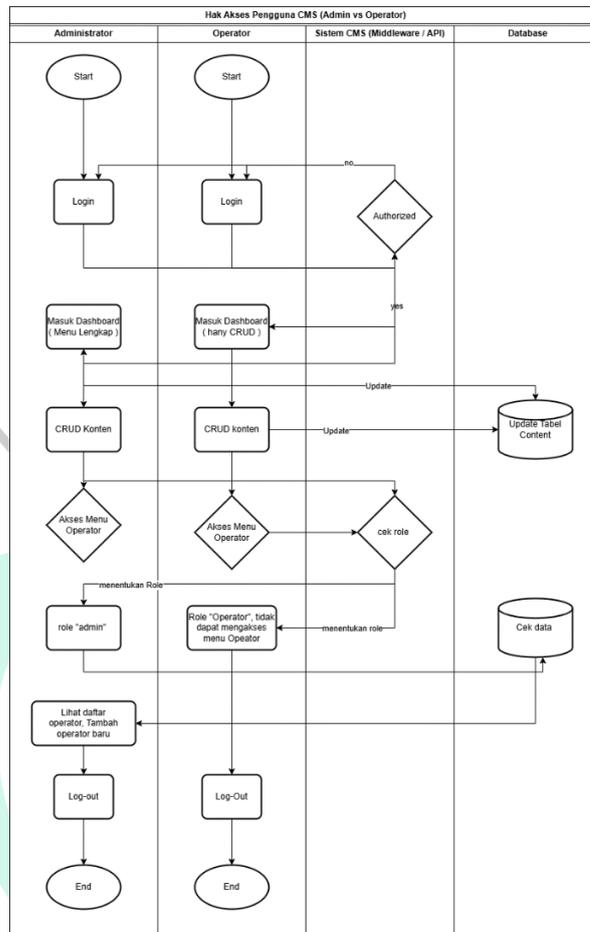
Pada **gambar 3.6** menampilkan *flowchart* pada sistem website yang sudah di perbarui. Dalam *flowchart* terlihat perbedaan dengan sistem yang lama. Jika sistem yang lama memerlukan tim developer untuk memperbarui isi konten jika ada perubahan. Pada sistem yang baru admin dan operator dapat membuat, melihat, mengedit isi konten yang ingin ditampilkan pada website tersebut. Sistem ini dinamakan CMS atau *Content Management System* yang memungkinkan terjadinya sistem *CRUD* atau *Creat, Read, Update, Delete* yang dapat dikerjakan oleh admin dan operator secara dinamis tanpa memerlukan bantuan tim developer. Hal ini menjadi

solusi agar kecepatan dalam memperbarui informasi bisa secara responsif dan juga cepat.



Gambar 3. 7 Flowchart sistem fetchData ke Front-end

Gambar 3.7 menunjukkan sebuah *flowchart* sistem, *flowchart* diatas menjelaskan bahwa alur kerja dari sistem fetch Data ke sisi *Front-end*. *Flowchart* digunakan untuk memvisualisasikan data yang dibutuhkan oleh pengguna antarmuka yang di kirim dari sisi server dapat ditampilkan pada tampilan depan. Dalam diagram diatas dijelaskan setiap tahapannya yang dilalui oleh permintaan data hingga ditampilkan seperti permintaan data yang dilakukan oleh *Front-end*, lalu diteruskan ke bagian *API* atau *controller*, yang kemudian mengambil data dari database, hingga akhirnya data tersebut dikirim kembali ke *Front-end* untuk ditampilkan kepada pengguna.

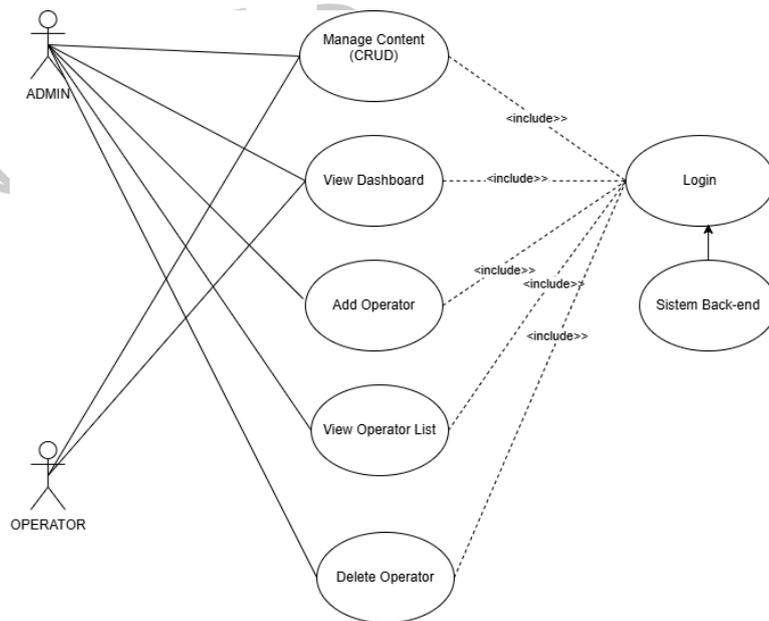


Gambar 3. 8 flowchart hak akses admin dan operator

Gambar 3.8, berfokus untuk menampilkan *flowchart* sistem untuk hak akses milik admin dan operator. Pada diagram diatas menunjukkan perbedaan antara admin dengan operator. Admin dan operator mendapatkan hak akses untuk memperbarui konten dan melihat isi konten yang sudah dibuat, namun perbedaannya adalah admin mendapatkan hak akses untuk melihat, membuat, dan menghapus operator yang ada, sedangkan operator tidak mendapatkan akses menu operator.

2. Membuat Use Case Diagram

Sistem yang dibangun didasarkan oleh analisis yang telah dilakukan dengan tujuan untuk mengelola konten pada website milik PT Bank Abdi. Sistem ini dibangun dengan model berorientasi objek dengan menggunakan *unified Modeling Language(UML)*. Di bawah ini merupakan 2 usecase yang ditampilkan.

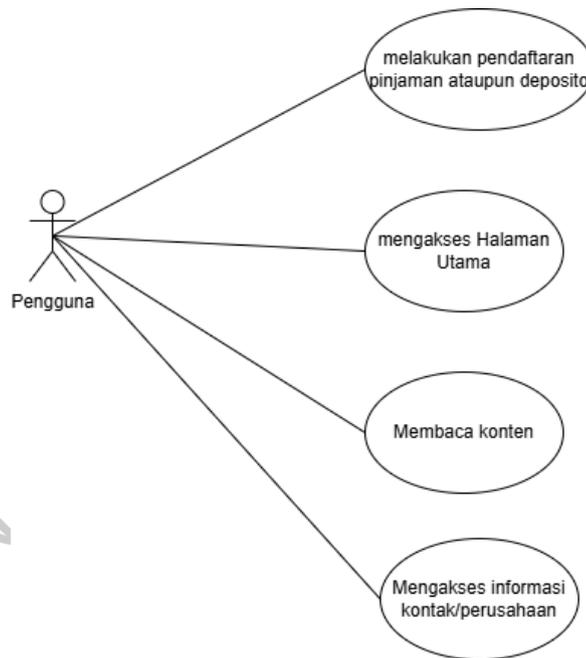


Gambar 3. 9 UsecaseDiagram admin dan operator

Dari usecase diatas, menampilkan usecase hak admin dan operator. Untuk lebih lengkapnya, berikut user *role matrix* :

No	Hak akses	Role	
		Admin	Operator
1	Melihat Dashboard	v	v
2	Melakukan Tambah Konten	v	v
3	Melakukan Edit Konten	v	v
4	melihat List Isi konten	v	v
5	Mengakses menu Operator	v	x
6	melihat List Operator	v	x
6	Menambahkan Operator	v	x

Tabel 3. 2 tabel user role matrix



Gambar 3. 10 usecase pengunjung website

3.2.3 Development

Pada tahap ini, penulis sudah mulai melakukan coding untuk membangun website milik pt bank abdi. Sesuai dengan *timeplan* langkah pertama yaitu slicingUI.

1. SlicingUI

Pada tahap ini Penulis sebagai *Full-Stack Developer* memulai dengan men-slicingUI milik website PT bank abdi. SlicingUI adalah sebuah kegiatan untuk mengkonversi tampilan design website dari desain yang diberikan. Namun karena penulis me reframe dari design website yang sudah ada, tidak ada perubahan signifikan dari tampilan website sebelumnya. Penulis memakai bahasa pemrograman JavaScript dengan *framework* Next.JS. jumlah halaman yang di slicing sekitar berjumlah 25 halaman website yang masing masing terhubung secara dinamis satu sama lain dengan menggunakan sistem yang diberikan oleh Next.Js yaitu *dynamic Routes* [id].tsx. contoh terapan penggunaan *dynamic routes* seperti di bawah ini.

```

allKontens.forEach(konten => {
  if (konten.sub_menu?.url && konten.status) {
    const urlParts =
konten.sub_menu.url.split('/');
    let routeId = urlParts.at(-1) ||
konten.sub_menu.url.replace(/^\/|^pinjaman\/g,
'');

    if (routeId) {
      const href = `/pinjaman/${routeId}`;
      backendMenuItems.push({
        href,
        label: konten.sub_menu.name ||
konten.sub_menu.sub_menu_name
      });
    }
  }
});

```

Fungsi *code* diatas dalam dynamic routes adalah **dynamic routes** berdasarkan data dari backend untuk ditampilkan di menu.

Pada tampilan UI yang sudah di slicing dengan bahasa pemrograman Next.Js. terdapat 6 menu pada website PT Bank Abdi yaitu menu Pinjaman, Tabungan, Deposito, Informasi, Tentang kami dan halaman Hubungi kami. 4 menu utama memiliki sub menunya masing masing yaitu

- **Menu: Pinjaman**
- SubMenu:
 - Kredit Modal Kerja
 - Kredit Investasi
 - Kredit Multiguna
 - Kredit Kepemilikan Rumah
 - Kredit Kepemilikan Mobil
 - Kredit Kendaraan Bermotor
 - Kredit Tanpa Agunan
 - Pengajuan Kredit (tidak masuk ke DB:subMenu)

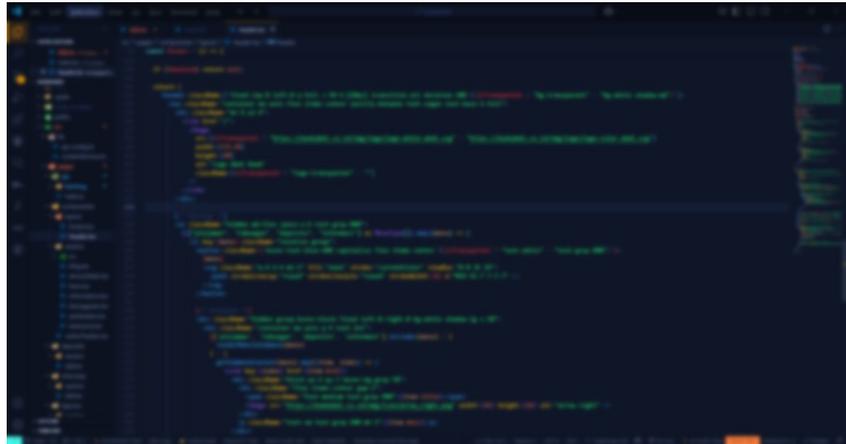
- **Menu: tabungan**
- SubMenu:
 - Tabungan Abdi
 - Tabungan Abdiku
 - Tabungan Abdi Simple
- **Menu: Deposito**
- SubMenu:
 - Deposito Berjangka
 - Formulir deposito (tidak masuk ke DB:subMenu)
 - Kalkulator deposito(tidak masuk ke DB:subMenu)
- Menu: Informasi
- SubMenu:
 - Laporan Tahunan/triwulan
 - Suku Bunga (tidak masuk ke DB:subMenu)

pada **gambar 3.11**, ini merupakan potongan *code* untuk halaman menu Pinjaman yang berisikan 7 sub menu dibawahnya, lalu pada **gambar 3.11**, merupakan *code* untuk *header.tsx* pada website



Gambar 3. 11 code menu pinjaman

Gambar 3.11 menampilkan *code* next.js untuk halaman menu pinjaman dimana terdapat fungsi dynamic routes yang dipakai. Dynamic routes berfungsi sebagai rute pada halaman menu pinjaman agar saling terintegrasi satu sama lain.



Gambar 3. 12 code header website

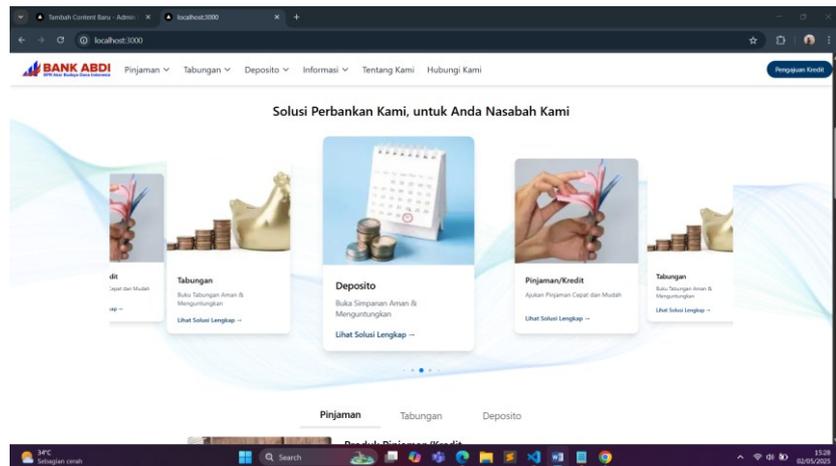
Gambar 3.12 menampilkan code milik header dari website PT Bank Abdi, header sendiri dipakai di seluruh halaman website PT Bank Abdi, sehingga membuat komponen `header.tsx` terpisah dari `code` utama. Alasannya adalah agar setiap komponen yang memakai header, tinggal memanggil props dari `code` header itu sendiri.

tidak terlalu berbeda dengan sebelumnya, dibawah ini merupakan tampilan website yang sudah di reframe dengan bahasa pemrograman yang baru:



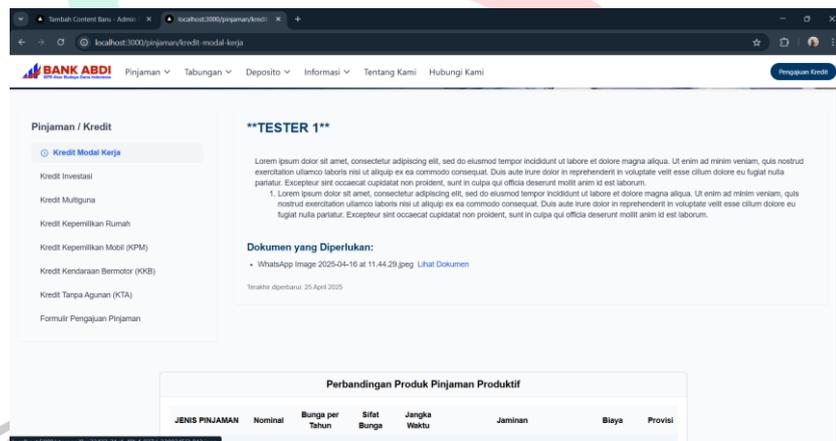
Gambar 3. 13 tampilan halaman utama

Pada **gambar 3.13** menampilkan halaman utama dari website PT Bank Abdi, ini merupakan halaman pertama yang akan di lihat oleh user ketika mengunjungi website milik PT Bank Abdi.



Gambar 3. 14 tampilan halaman utama (2)

Gambar 3.14 masih memberikan tampilan halaman utama jika di scroll kebawah, terdapat menu yang ditampilkan dalam bentuk card seperti menu pinjaman, tabungan dan deposito.

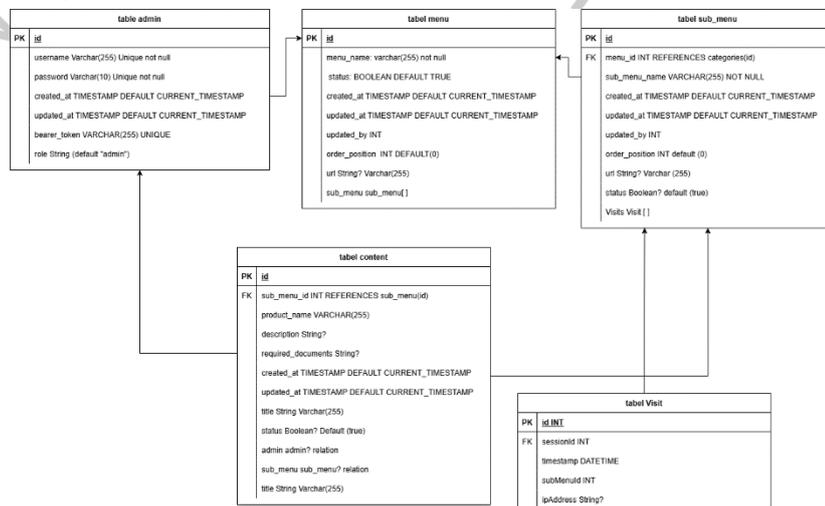


Gambar 3. 15 tampilan halaman Kredit Modal Kerja

Pada gambar 3.15 merupakan tampilan dari halaman kredit modal kerja, kredit modal kerja merupakan sub menu dari menu pinjaman. Administrator dan operator dapat memperbarui isi konten pada 12 sub menu yang sudah di sebutkan diatas.

2. Membuat ERD atau Entity Relationship Diagram untuk kebutuhan *Back-end Developer CMS*

ERD atau Entity relationship diagram merupakan sebuah data flow yang di gambarkan dengan bentuk sebuah diagram untuk menampilkan data store. Erd ini dapat membantu untuk memvisualisasikan bagaimana data ini saling terhubung dan berguna untuk menjadi sebuah arsitektur basis data (database) relasional. ERD yang saya pakai untuk membangun struktur *back-end* dalam proyek reengineering website PT BANK ABDI seperti di bawahini:



Gambar 3. 16 Entity Relationship Diagram

Gambar 3.16 menggambarkan struktur data dari website PT Bank Abdi. Struktur data tersebut terdiri dari nama Tabel, tipe data, method, dan relasi di setiap masing - masing Tabel.

3. Membuat sistem *back-end CMS*

Sistem CMS atau *Content Management System* untuk membantu dalam mengupdate informasi serta konten pada website dibangun dengan bahasa pemrograman Next.JS. semua sistem *back-end* dibangun menggunakan bahasa Next.js serta memakai ORM atau *Object-Relational Mapping* Prisma Next.js.

Prisma Next.js adalah sebuah sistem *Object-Relational Mapping* yang mempermudah dalam Mengakses dan

memanipulasi data dari API. Cara kerja dari prisma next js adalah menulis kueri database dengan cara yang lebih efisien dan menghindari penulisan SQL secara mentah. Prisma juga berperan sebagai connector dari database ke sistem dan mendefinisikan model data dengan file `schema.prisma`. contoh prisma yang digunakan di dalam sistem *back-end* website bank abdi seperti pada dibawah ini:



Gambar 3. 17 code `schema.prisma`

Cara mengaplikasikannya adalah dengan cara menulis *code*:

```
// pages/api/users.ts
import { PrismaClient } from '@prisma/client';

const prisma = new PrismaClient();

export default async function handler(req, res)
{
  const users = await prisma.user.findMany();
  res.status(200).json(users);
}
```

Hasilnya adalah ketika mengakses endpoint <http://localhost:3000/api> maka fungsi handler akan dijalankan dan akan mendapatkan *response* JSON berisi seluruh data dari tabel yang di panggil oleh sistem *Front-end*.

Content Management System adalah sebuah sistem yang memudahkan administrator untuk memperbarui isi dari sebuah konten atau halaman website. CMS terdiri dari 2 elemen yaitu CMA dan CDA. CMA atau *Content Management Application* berfungsi untuk menangani pengelolaan konten dengan mudah, biasanya ditampilkan dalam *Front-end* dengan editor **WYSIWYG (What You See Is What You Get)**. CDA adalah *Content Delivery Application* adalah bagian *back-end*nya atau lebih mudahnya adalah bagian dapur dari websitenya. CDA berfungsi untuk mempublikasikan lewat CMA agar dapat di lihat oleh visitor pengunjung. Contoh *code* dan tampilan nya adalah seperti di bawah ini

```
import dynamic from "next/dynamic";
import "react-quill-new/dist/quill.snow.css";
interface DescriptionFormatSelectorProps {
  konten: string;
  setKonten: (konten: string) => void;
}

const ReactQuill = dynamic(() => import("react-quill-new"), { ssr: false });

const modules = {
  toolbar: [
    [{ font: [] }, { size: [] }],
    ["bold", "italic", "underline"],
    [{ list: "ordered" }, { list: "bullet" }],
    [{ align: [] }],
    ["link"],
    ["clean"],
    [{ indent: "-1" }, { indent: "+1" }],
  ],
};

const formats = [
  "font", "size", "bold", "italic", "underline",
  "list", "align", "link", "indent"];

const DescriptionFormatSelector = ({ konten,
setKonten }: DescriptionFormatSelectorProps) =>
{
  return (
    <div className="w-full p-4 border rounded-md
bg-white">
```

```

    <h2 className="text-black text-sm font-
semibold mb-2">Tulis Deskripsi</h2>
    <ReactQuill
      value={konten}
      onChange={setKonten}
      modules={modules}
      formats={formats}
      className="text-black"
    />
  </div>
);
};

export default DescriptionFormatSelector;

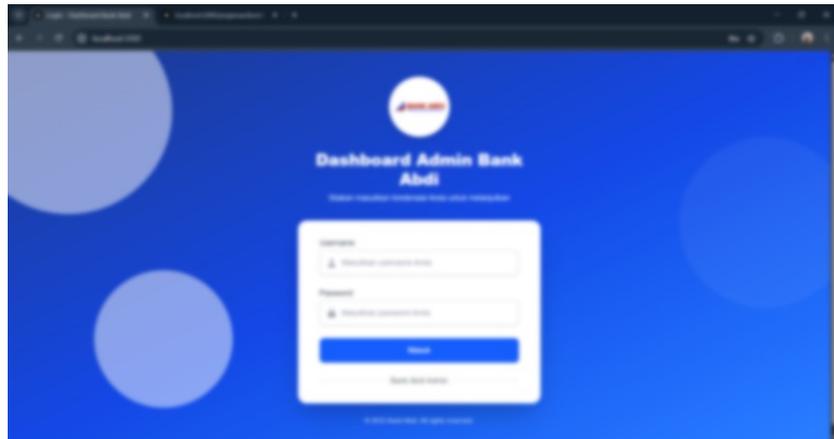
```

Code diatas adalah penggalan komponent pada code DescriptionFormatSelector.tsx berfungsi untuk memberikan tampilan CMA kepada editor agar editor dapat menulis isi konten atau memperbarui isi konten pada website, tampilan pada *Front-end* adalah seperti dibawah ini



Gambar 3. 18 tampilan halaman tambah konten

Pada **gambar 3.18** merupakan tampilan pada halaman menambahkan konten baru pada website. Selain pada halaman tambah konten juga ada beberapa halaman pada dashboard admin pada *back-end* seperti:



Gambar 3. 19 tampilan menu login

Pada **gambar 3.19** menampilkan tampilan menu login, yang dapat login hanya admin dan operator yang terdaftar di sistem, contohnya seperti pada gambar 3.11 admin dengan nama reyki, serta operator dengan nama femas, dan andi. Admin dan operator memasukkan username mereka dan password mereka untuk dapat masuk ke halaman dashboard.



Gambar 3. 20 tampilan halaman dashboard admin

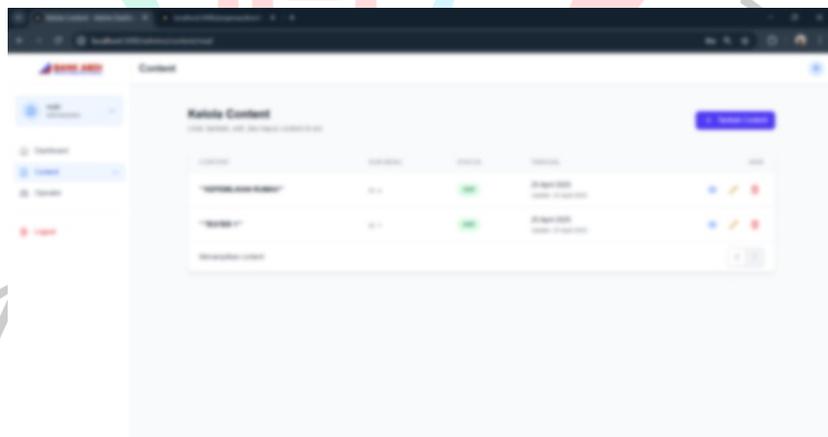
Gambar 3.20 menampilkan halaman dashboard, disini terdapat beberapa menu yaitu konten (terdapat 2 sub menu create konten, dan read konten) dan menu operator. Serta terdapat grafik dimana admin dan operator dapat melihat grafik *visitor* yang berkunjung ke website PT Bank Abdi. Dalam grafik visitor dibuat untuk sesi perbulan.

Sistem tracking visitor ini berfungsi agar PT Bank abdi mengetahui jumlah pengunjung websitenya dalam perbulan serta halaman mana saja yang paling sering di kunjungi.



Gambar 3. 21 halaman tambah konten

Pada **gambar 3.21** menampilkan halaman *create* tambah konten baru. Admin dan operator dapat menulis konten baru di halaman ini.



Gambar 3. 22 tampilan list konten

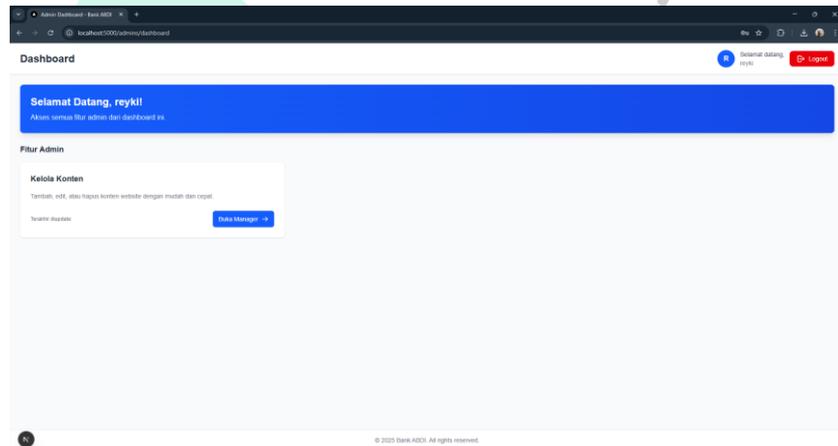
Gambar 3.22 menampilkan list konten yang sudah dibuat pada halaman *create*, semua konten akan ditampilkan pada halaman ini. Admin dan operator dapat mengedit, melihat serta menghapus konten yang ada, semua berada di halaman ini.

3.2.4 Testing

1. Menguji kelayakan website dan sistem CMS

Pada tahap ini, penulis sudah melakukan integrasi antara *Front-end* dan juga *back-end*. Setelah semua ter-integrasi, website dan sistem di uji apakah ada bug atau ada kesalahan pada website dan sistem *back-end*. Pengujian dilakukan secara intens oleh *PM (Project Manager)*, *Leader Front-end* dan juga *Leader Back-end*.

Uji kelayakan website dan sistem terjadi beberapa kali, karena sistemnya adalah dengan *check-point* yang dilakukan, terdapat beberapa kali perubahan mulai dari tampilan dan juga sistem seperti dibawah ini:



Gambar 3. 23 Tampilan Dashboard versi Beta

Gambar 3.23 menampilkan tampilan awal dari dashboard admin versi lama, yang digunakan sebelum dilakukan berbagai proses evaluasi dan pengujian. Setelah melalui serangkaian uji kelayakan, *checkpoint*, serta pengumpulan masukan dari pengguna dan tim pengembang, tampilan antarmuka website serta sistem *back-end* mengalami sejumlah revisi dan penyempurnaan. Proses iteratif tersebut menghasilkan pembaruan signifikan, baik dari sisi fungsionalitas maupun desain, yang kemudian diwujudkan dalam versi terbaru seperti yang ditampilkan dalam gambar selanjutnya.



Gambar 3. 24 Tampilan Dashboard versi Final

Tampilan pada **Gambar 4.2** merupakan hasil akhir dari proses pengembangan, yang menampilkan tampilan final dari dashboard admin. Desain ini telah melalui berbagai tahapan revisi, evaluasi fungsionalitas, serta penyempurnaan tampilan antarmuka berdasarkan hasil uji kelayakan dan masukan dari pengguna. Dashboard ini mencerminkan versi final yang siap digunakan secara optimal dalam mendukung kebutuhan pengelolaan konten dan administrasi sistem

3.2.5 Feedback

Penulis telah melakukan kegiatan presentasi kepada pihak manajemen PT Bank Abdi sebagai bagian dari tahapan evaluasi dan pelaporan hasil pengembangan proyek yang telah dikerjakan. Presentasi ini dilaksanakan pada hari Jumat, tanggal 25 April 2025, pukul 10.00 WIB, bertempat di kantor pusat PT Bank Abdi. Kegiatan tersebut dihadiri oleh sejumlah perwakilan penting dari pihak perusahaan, yaitu Direktur Utama Ibu Wardati, S.E., Direktur Operasional Bapak Mario Yahya, serta tim IT Support PT Bank Abdi yang diwakili oleh Bapak Gunawan. Dalam kegiatan ini, penulis tidak melakukan presentasi secara individu, melainkan didampingi oleh Project Manager proyek yang bersangkutan, yaitu Bapak Andi, yang turut membantu menjelaskan berbagai aspek teknis dan strategis dari sistem yang telah dikembangkan.

Tujuan utama dari presentasi ini adalah untuk menyampaikan hasil akhir proyek kepada pihak perusahaan, termasuk menjelaskan perubahan-perubahan signifikan yang telah dilakukan selama proses pengembangan serta fitur-fitur utama yang telah berhasil diimplementasikan. Selain sebagai bentuk pertanggungjawaban, presentasi ini juga berfungsi sebagai sarana untuk memperoleh umpan balik langsung dari stakeholder utama, guna mengetahui sejauh mana sistem yang dibangun telah memenuhi kebutuhan dan ekspektasi pengguna. Selama sesi diskusi, pihak PT Bank Abdi memberikan sejumlah masukan konstruktif yang berkaitan dengan tampilan antarmuka website, performa sistem, serta aspek fungsional lainnya. Masukan-masukan tersebut sangat berguna sebagai dasar untuk perbaikan dan pengembangan sistem ke depannya. Dokumentasi kegiatan presentasi ini dapat dilihat pada foto yang terlampir dalam lampiran laporan.

3.2.6 Performa Website

Performa website merupakan salah satu indikator utama dalam menilai keberhasilan suatu situs web, terutama dalam konteks layanan perbankan seperti PT Bank-Abdi yang menuntut kecepatan, stabilitas, dan efisiensi. Pada proyek *reframe* dan *reengineering* ini, peningkatan performa website menjadi fokus utama melalui penerapan teknologi modern yaitu Next.js dan integrasi sistem *Content Management System (CMS)*.

Next.js dipilih karena kemampuannya dalam mendukung *Server-side Rendering (SSR)* dan *Static Site Generation (SSG)* yang secara langsung berdampak pada peningkatan kecepatan load halaman dan pengalaman pengguna (*user experience*). Dengan SSR, halaman dapat di-render di server sebelum dikirim ke browser, sehingga waktu akses pertama menjadi lebih cepat. Selain itu, optimasi bawaan seperti *image optimization*, *code splitting*, dan *caching* mendukung performa secara menyeluruh.

Sementara itu, integrasi CMS memungkinkan tim non-teknis untuk melakukan pengelolaan konten secara mandiri tanpa harus melakukan deploy ulang. Ini mempercepat waktu pembaruan konten dan mengurangi beban developer. Hasilnya, proses pengelolaan website menjadi lebih efisien, fleksibel, dan terorganisir.

3.3 Kendala Yang Dihadapi

Selama menjalani masa kerja profesi selama tiga bulan di PT ASLI Rancangan Indonesia, penulis menghadapi beberapa kendala. Salah satu kendala yang dihadapi adalah kesalahan dalam penulisan kode. Penulis diminta untuk menerapkan prinsip *clean code*, yaitu menulis kode secara rapi dan terstruktur, dengan memisahkan fungsionalitas ke dalam beberapa komponen agar sistem lebih mudah dipahami dan dikelola.

Selain itu, penulis juga mengalami kesulitan dalam mencari aset-aset yang diperlukan untuk pembuatan ulang (*reframe*) website. Penulis tidak mendapatkan akses desain dari Figma sehingga harus mengandalkan tampilan dari website lama yang sudah di-hosting serta menggunakan *source code* sebelumnya sebagai acuan dalam pengembangan ulang.

Penulis juga mengalami kesulitan saat mengerjakan sistem back-end, khususnya dalam proses pembuatan *Entity Relationship Diagram* (ERD). Kendala utama yang dihadapi adalah penulis tidak melakukan *checkpoint* atau validasi terlebih dahulu dengan *Project Manager* maupun tim back-end setelah menyelesaikan ERD. Akibatnya, terjadi beberapa revisi yang membuat penulis harus memperbarui skema Prisma (Next.js) serta menyesuaikan kembali tabel-tabel yang telah dibuat sebelumnya. Perubahan tersebut menyebabkan beberapa gangguan, seperti munculnya error pada sistem karena struktur yang belum sinkron.

Dalam beberapa kasus, data yang sudah ada hilang saat proses *migration* dilakukan setelah perubahan diterapkan. Selain masalah dalam membangun sistem front-end dan back-end, penulis juga mendapatkan kesulitan membangun logika pemrograman yang digunakan di perusahaan, karena banyaknya *library* yang mendukung, penulis tidak

mengetahui untuk mempercepat pembangunan sistem dan website harus memakai *library* apa saja.

3.4 Cara Mengatasi Kendala

Dalam menghadapi kendala yang dihadapi selama bekerja di perusahaan, penulis melakukan beberapa cara untuk mengatasi kendala yang dihadapi, hal ini juga melatih penulis untuk siap dalam menghadapi kendala yang ada, sebagai bekal untuk kedepannya didalam dunia kerja profesional. Ada beberapa cara seperti:

1. Penulis berkonsultasi kepada staf di dalam kantor bagaimana cara menghadapi kendala yang terjadi. Senior serta staf mengajarkan untuk memulai langkah-langkah seperti yang di minta.
2. Penulis melakukan *check-point* saat per-modul yang dikerjakan selesai. Hal ini dilakukan agar *lead* dan PM mengetahui progress yang sedang penulis kerjakan.
3. Penulis memakai beberapa *library* pada React untuk membantu mempercepat building tampilan website, dan penulis melakukan riset untuk mengetahui teknologi dan *engine* apa saja yang cocok dalam membangun sistem back-endnya agar terlihat maksimal.

3.5 Pembelajaran Yang Diperoleh dari Kerja Profesi

Selama menjalani kegiatan Kerja Profesi di perusahaan PT ASLI Rancangan Indonesia (ASLI RI), penulis banyak sekali mendapatkan pembelajaran serta pengalaman yang didapatkan. Posisi penulis sebagai *Full-Stack Developer* pada bagian tim *IT Developer*, penulis banyak mendapatkan masukan, pembelajaran mengenai teknologi yang digunakan, sistem yang digunakan dalam dunia kerja nyata terutama di dunia *professionalisme*.

Penulis menyadari pentingnya pendekatan yang berfokuskan kepada pembuatan website yang dinamis serta responsive, namun di dukung oleh teknologi yang dipakai agar hasil yang diharapkan menjadi lebih baik lagi dan bertahan lama. Kerja sama tim juga di perlukan, selain mengerjakan proyek reframe dan reengineering website milik PT Bank Abdi, penulis juga turut ikut serta dalam membantu beberapa proyek milik PT ASLI Rancangan Indonesia

Hal ini juga memberikan pembelajaran kepada penulis betapa pentingnya berkomunikasi satu sama lain dengan tim lain seperti tim QA atau Quality Assurance terkait kelayakan pada sistem yang dibuat. Komunikasi dalam dunia *IT Developer* menjadi kunci utama untuk menyelesaikan proyek-proyek yang diberikan. Tidak ada tim IT yang bekerja sendiri karena semua bergantung kepada kerja sama tim, mulai dari tim *Front-end Developer*, *Back-end Developer*, *Quality Assurance*, dan lain-lain.

Dari seluruh rangkaian pengalaman kerja selama kegiatan Kerja Profesi yang dijalankan, Penulis lebih mendalami pemahaman mengenai cara membangun suatu website, meng-integrasi sistem front-end dengan sistem server atau back-end serta mempelajari apa itu *Server Side Rendering*, *Client Side Rendering* untuk memaksimalkan optimalisasi sebuah website yang dinamis.

Dalam hal ini, penulis menjadikan pengalaman serta pembelajaran yang diberikan sebagai kesempatan untuk mengembangkan diri dalam mempersiapkan diri untuk lebih siap kedepannya dalam dunia profesional, terutama dalam dunia *IT Developer*