README.md

NextJS Tingkat Lanjut

Nama	Ahmad Mu'min Faisal
NIM	1203210101
Kelas	IF-01-03

1 Persiapan

Tugas ini melanjutkan dari modul 5, yaitu NextJS Tingkat Dasar. Tetapi, sebelum melanjutkan ke kode program, install `axios` terlebih dahulu agar bisa melakukan HTTP request.

```
npm install axios
```

2 Source Code Tambahan

2.1 src/app/posts/page.js

Source code yang diberikan adalah bagian dari halaman Next.js yang bertujuan untuk menampilkan daftar artikel (`posts`). Pertama, library yang diimport termasuk axios untuk melakukan HTTP requests, Link dari Next.js untuk menangani navigasi antar halaman, dan fungsi `getArticle` yang dipanggil untuk mengambil data artikel dari layanan utility yang terdapat di dalam folder `utils/services/post.service` .

Fungsi utama `Page` adalah sebagai fungsi asynchronous yang akan dieksekusi saat halaman ini di-load. Dalam fungsi ini, data artikel diambil menggunakan `getArticle()` dan hasilnya di-destructure untuk mendapatkan properti data.

Selanjutnya, halaman ini merender konten dengan menampilkan judul halaman dan daftar artikel. Setiap artikel ditampilkan sebagai link yang mengarah ke halaman detail artikel dengan menggunakan Link dari Next.js. Gaya tampilan halaman diatur menggunakan elemen HTML dan Tailwind CSS untuk mengatur ukuran font, warna teks, dan efek hover.

File `src/app/posts/page.js` :

```
import axios from "axios";
import Link from "next/link";
import { getArticle } from "@/utils/services/post.service";
export default async function Page() {
  // const { data } = await axios.get("https://raw.githubusercontent.com/d
  const { data } = await getArticle();
  return (
    <div>
      <h1 className="font-bold text-2xl text-gray-600">Posts</h1>
      <div className="py-4 text-gray-600">
        {data.map((post) => (
          <>
            <div className="p-4 my-3">
              <Link href={`/posts/${post.id}`}>
                <b className="hover:text-blue-400">{post.title}</b>
              </Link>
            </div>
          </>
        ))}
      </div>
    </div>
  );
}
```

2.2 src/app/posts/[slug]/loading.js

Komponen ini dibuat untuk menampilkan pesan "Ini Loading". Dalam konteks proyek Next.js, file ini memiliki lokasi di dalam folder src/app/posts/[slug]/, menunjukkan kemungkinan penggunaannya dalam halaman detail artikel yang mungkin memuat data dinamis berdasarkan slug.

Dalam struktur komponen, hanya terdapat sebuah elemen `<h2>` yang berisi teks "Ini Loading". Tampilannya sederhana dan bertujuan memberikan umpan balik visual kepada pengguna bahwa aplikasi sedang dalam proses pemuatan data atau melakukan tugas tertentu sebelum menampilkan halaman lengkapnya. Komponen ini diekspor sebagai default dari file, memungkinkan penggunaannya di berbagai bagian proyek dengan cara yang mudah. Keseluruhannya, komponen ini dapat digunakan untuk meningkatkan pengalaman pengguna dengan memberikan indikasi visual bahwa proses pemuatan sedang berlangsung.

File `src/app/posts/[slug]/loading.js` :

```
const Loading = () => {
  return (
    <>
```

```
      <h2>Ini Loading</h2>
    </>
  );
};
export default Loading;
```

2.3 src/app/posts/[slug]/page.js

Dalam file ini, terdapat sebuah fungsi default yang disebut Page. Fungsi ini menerima properti params, yang berasal dari parameter dinamis [slug] yang digunakan dalam struktur path URL. Fungsi ini berfungsi untuk menampilkan halaman detail artikel berdasarkan slug yang diterima.

Pertama, di-import dua modul yaitu `notFound` dari `next/navigation` dan fungsi `getArticleDetail` dari layanan utilitas yang terdapat di dalam folder `utils/services/post.service`. Modul `notFound` dari Next.js digunakan untuk menangani kondisi ketika data artikel tidak ditemukan, sedangkan `getArticleDetail` digunakan untuk mengambil detail artikel berdasarkan slug.

Setelah itu, terdapat sebuah pengecekan data dengan menggunakan `if (!data)` yang mengindikasikan bahwa jika data artikel tidak ada, maka halaman akan diarahkan ke halaman 404 menggunakan fungsi `notFound()`.

Jika data artikel berhasil diambil, halaman akan merender elemen HTML yang menampilkan judul artikel dan isi kontennya. Judul artikel ditampilkan dalam elemen `<h1>`, dan kontennya ditampilkan dalam elemen `<p>`. Penggunaan properti `data.title` dan `data.content` menunjukkan bahwa objek data yang diambil memiliki struktur yang memuat informasi judul dan konten artikel.

File `src/app/posts/[slug]/page.js`:

```
import { notFound } from "next/navigation";
import { getArticleDetail } from "@/utils/services/post.service";
export default async function Page({ params }) {
  // const { data } = await new Promise(async (resolve) => {
  //     setTimeout(() => {
  //         // resolve('Ini Data dari menunggu');
  //         resolve(false);
  //     }, 3000);
  // });
  const { data } = await getArticleDetail(params.slug);
  if (!data) {
    notFound();
  }
  return (
    <div>
      <h1 className="font-bold text-2xl text-gray-600">Posts - {data.title
```

```
        <div className="py-4 text-gray-600">
          <p>{data.content}</p>
        </div>
      </div>
    );
  }
```

2.4 src/app/not-found.js

File ini dibuat untuk menangani error halaman tidak ditemukan di route `root`.

File `src/app/not-found.js`:

```
export default function NotFound() {
  return (
    <div>
      <h2>Ini 404! Global</h2>
    </div>
  );
}
```

2.5 src/app/posts/[slug]/not-found.js

File ini dibuat untuk menangani error halaman tidak ditemukan di route `posts`.

File `src/app/posts/[slug]/not-found.js`:

```
export default function NotFound() {
  return (
    <div>
      <h2>Ini 404! Post</h2>
    </div>
  );
}
```

2.6 src/middleware.js

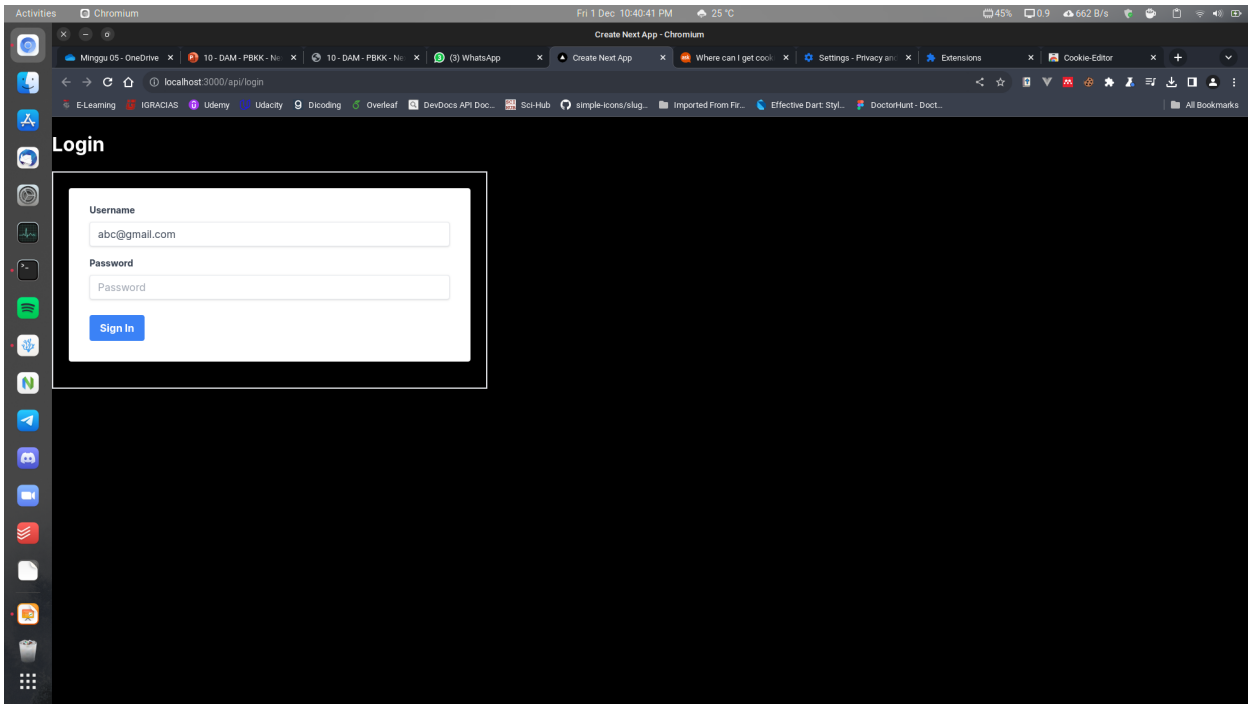
Kode yang diberikan adalah middleware untuk server dalam proyek Next.js. Middleware ini bertujuan untuk melakukan pengecekan otorisasi sebelum memberikan akses ke halaman atau data dengan path `/posts`. Pertama, middleware memeriksa keberadaan cookie `accessToken` dalam permintaan. Jika cookie tersebut ditemukan, otorisasi dianggap berhasil, dan pengguna diizinkan melanjutkan. Jika tidak, pengguna akan diarahkan ke halaman login melalui URL redirect. Konfigurasi dengan properti matcher: `/posts` menunjukkan bahwa middleware ini hanya diterapkan pada URL dengan pola `/posts`. Keseluruhan, middleware ini memberikan kontrol akses berdasarkan keberadaan token otorisasi sebelum mengizinkan pengguna untuk mengakses halaman atau data tertentu.

File `src/middleware.js` :

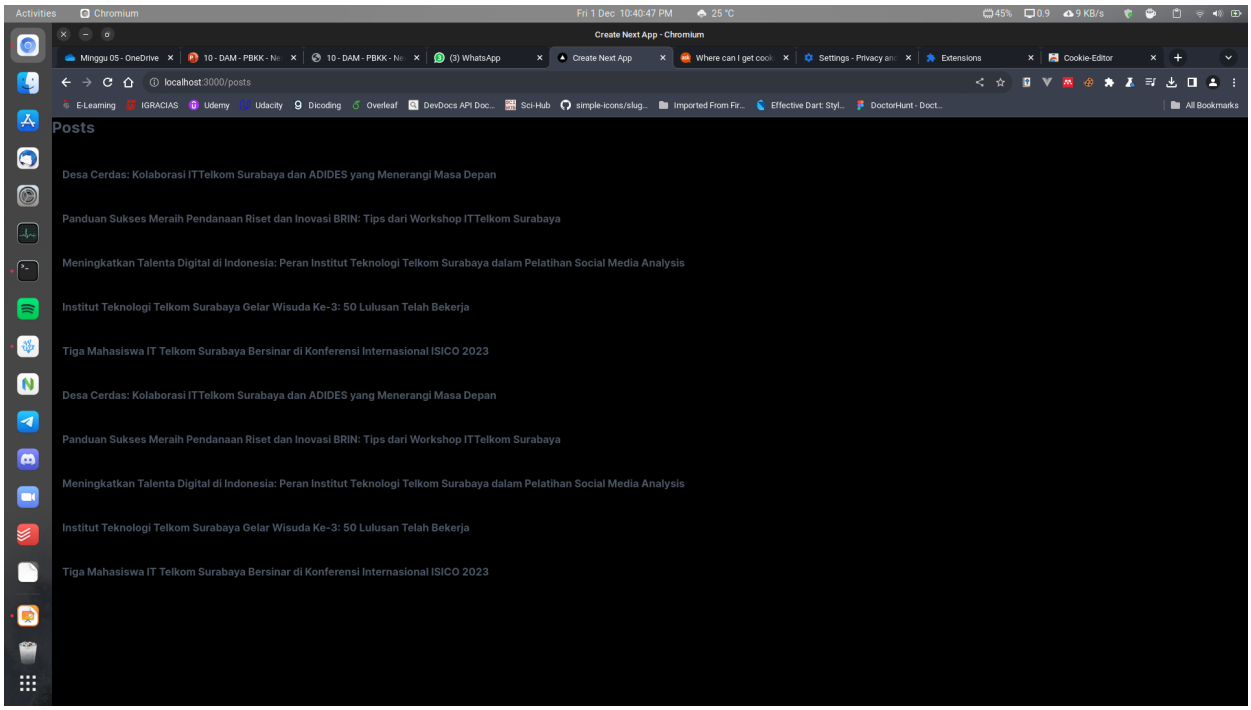
```
import { NextResponse } from "next/server";
export function middleware(request) {
  let checkAuth = false;
  const checkToken = request.cookies.get("accessToken");
  if (checkToken) {
    checkAuth = true;
  }
  if (checkAuth) {
    return NextResponse.next();
  } else {
    const loginUrl = new URL("/api/login", request.url);
    return NextResponse.redirect(loginUrl);
  }
}
export const config = {
  matcher: "/posts",
};
```

3 Dokumentasi

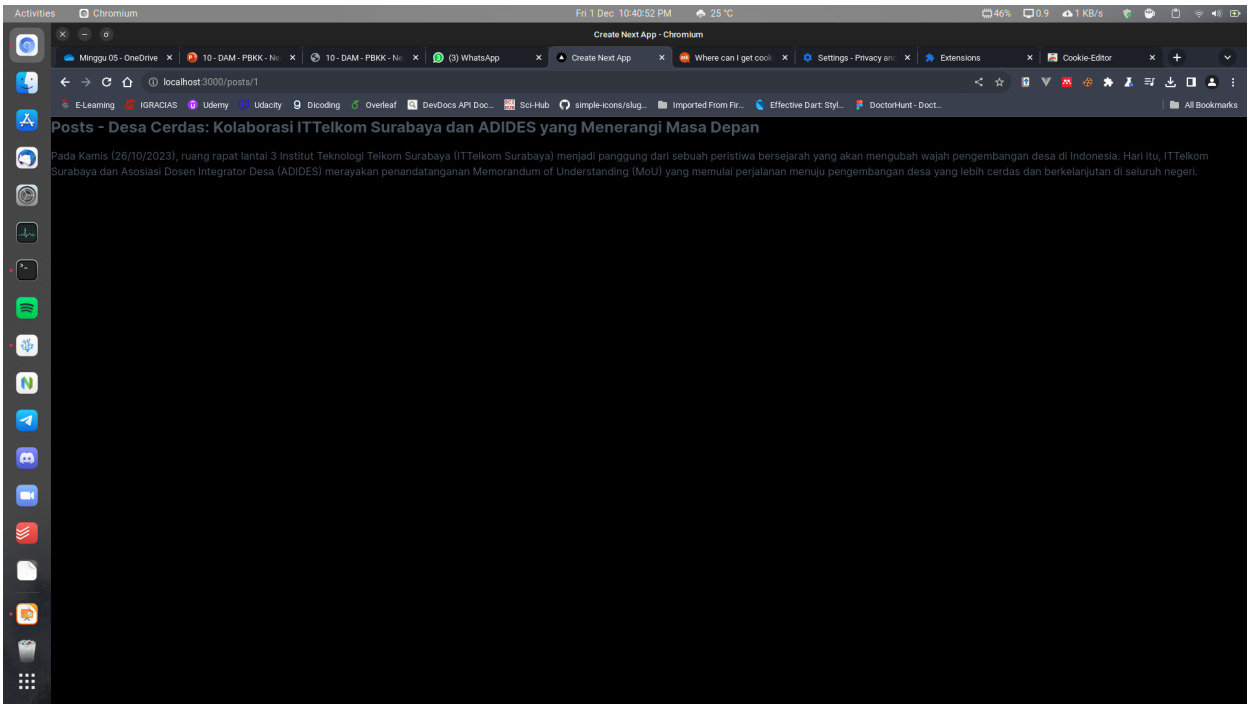
3.1 Halaman Login



3.2 Halaman Daftar Article



3.3 Halaman Detail Article


Activities
Chromium
Fri 1 Dec 10:40:52 PM
25 °C
46%
0.9
1 KB/s
Create Next App - Chromium
Minggu 05 - OneDrive
10 - DAM - PBKK - Ne
10 - DAM - PBKK - Ne
(3) WhatsApp
Create Next App
Where can I get cook
Settings - Privacy an
Extensions
Cookie-Editor
localhost:3000/posts/1
E-Learning
IGRACIAS
Udemy
Udacity
Dicoding
Overleaf
DevDocs API Doc...
Sci-Hub
simple-icons/slug...
Imported From Fir...
Effective Dart: Styl...
DoctorHunt - Doct...
All Bookmarks
Posts - Desa Cerdas: Kolaborasi ITTelkom Surabaya dan ADIDES yang Menerangi Masa Depan
Pada Kamis (26/10/2023), ruang rapat lantai 3 Institut Teknologi Telkom Surabaya (ITTelkom Surabaya) menjadi panggung dari sebuah peristiwa bersejarah yang akan mengubah wajah pengembangan desa di Indonesia. Hari itu, ITTelkom Surabaya dan Asosiasi Dosen Integrator Desa (ADIDES) merayakan penandatanganan Memorandum of Understanding (MoU) yang memulai perjalanan menuju pengembangan desa yang lebih cerdas dan berkelanjutan di seluruh negeri.