

Nama : NARINDA GENTA ROSASIA

NIM : 320200401019

PRAKTIKUM WEB DEVELOPMENT

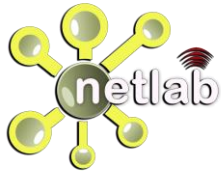
Minggu ke-9, Sesi ke-1

Case Study

Github : https://github.com/narindagenta/Modul20_narinda

- Setelah menginstal dan dapat menjalankan, silakan akses /pages/login.js. Kalian akan melakukan menambahkan kode untuk melakukan hit endpoint dari backend yang sudah dibuat sebelumnya pada sesi sebelumnya. Pastikan backend kalian berjalan.
- Hanya tambahkan kode yang diminta, biarkan yang lain seperti biasa.
- Jika kode sudah berjalan dengan baik dan sudah tersambung dengan backend, kode akan secara otomatis akan memindahkan halaman dari /login ke /profile

```
const email = localStorage.getItem('');  
axios.post()  
localStorage.setItem('user', user)  
await axios.post('http://localhost:5000/login', {  
  email: data.get('email'),  
  pass: data.get('password'),  
}).then((res) => {  
  alert('login sukses')  
  console.log(res)  
  localStorage.setItem('user', JSON.stringify(res.data.user))  
  localStorage.setItem(token, res.data.toke)  
  window.location.href = '/profile'  
}).catch((err) => {  
  console.log(err)  
  alert('login gagal:' + err.response.data.message)  
})  
}
```



- Selanjutnya, silakan buka file /pages/register. Lakukan hal yang sama dengan melakukan register ke endpoint yang sudah dibuat pada sesi sebelumnya.

```
const Register = () => {
  const handleSubmit = async (event) => {
    event.preventDefault();
    const data = new FormData(event.currentTarget);
    axios.post('http://localhost:5000/api/users/register', {
      name: data.name,
      email: data.email,
      password: data.password}, {
      headers: {
        'Content-Type': 'application/json'
      })
      .then((res) => {
        console.log("server response:", res);
      })
      .catch((err) => {
        console.log("Server responded with error", err);
      })

    console.log({
      username: data.get('username'),
      pass: data.get('password'),
      email: data.get('email'),
    })
  }
}
```

```
const email = localStorage.getItem('email');
axios.post()
localStorage.setItem('user', user)
await axios.post('http://localhost:5000/login', {
  email: data.gen('email'),
  pass: data.get('password'),
}).then((res) => {
  alert('login sukses')
  console.log(res)
  localStorage.setItem('user', JSON.stringify(res.data.user))
  localStorage.setItem('token', res.data.token)
  window.location.href = '/profile'
}).catch((err) => {
  console.log(err)
  alert('login gagal: ' + err.response.data.message)
})

// Tambahkan kode di bawah ini untuk mengambil data dari localStorage
// 1. Lakukan Axios POST ke backend pada endpoint /login di bawah ini,
// dengan parameter 'email' dan 'pass' yang didapat dari form (clue ada pada line 23 dan 24).
// simpan 'token' dan 'user' ke localStorage
```

DEBUG CONSOLE TERMINAL JUPYTER



- Jika kode berhasil dan sukses, kalian akan dibawa ke halaman /login.

A screenshot of a web browser window displaying a login page. The browser's address bar shows the URL "http://localhost:3000/login". The page has a dark header with various icons and a clock showing "Fajr: 03:48:35 Remaining 04:10:57". The main content area is white and titled "Sign in". It contains two input fields: "Email Address *" with the text "narinda" and "Password *". Below these is a checkbox labeled "Remember me". A blue "SIGN IN" button is positioned below the checkbox. At the bottom, there is a link that says "Don't have an account? Sign Up".

Sign in

Email Address *
narinda

Password *

☐ Remember me

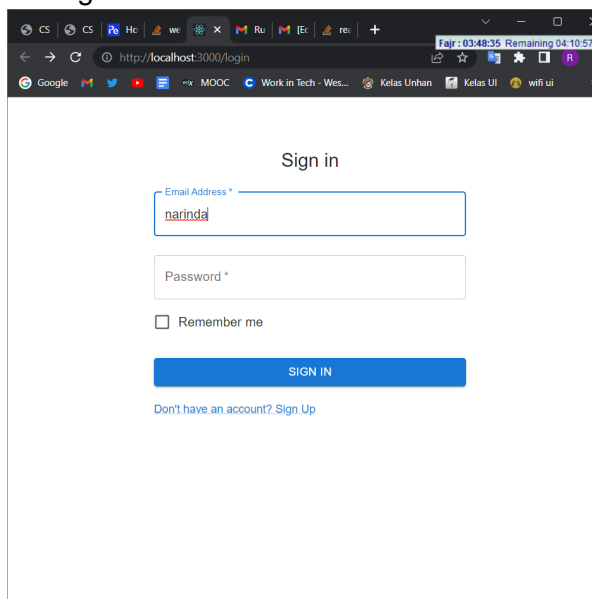
SIGN IN

[Don't have an account? Sign Up](#)

- Buka /pages/profile.js. Silakan lakukan hal yang sama. Satu hal yang membedakan di sini adalah kode akan melakukan hit endpoint ke /verify dengan tujuan untuk mengecek apakah token yang disimpan sama seperti data yang disimpan pada localStorage. Jangan lupa untuk melakukan setUser supaya data yang ditampilkan adalah data yang disimpan pada localStorage.

```
// Modifikasi kode di bawah ini untuk mengambil data dari localStorage
React.useEffect(() => {
  // 1. Ambil data user dari localStorage
  const user = localStorage.getItem('user')
  const token = localStorage.getItem('token')
  // 2. buat fungsi verifikasi token yang sama seperti di halaman home
  const verifyToken = async () => {
    await axios.post('http://localhost:3000/verify', {
      jwt: token
    }).then((res) => {
      if(res.status === 200 && res.data.id === JSON.parse(user).id) {
        console.log('token valid')
        setIsLogin(true)
      } else {
        alert('Token Salah')
        window.location.href = '/login'
      }
    }).catch((err) => {
      console.log(err)
      alert('token kadaluarsa')
      window.location.href = '/login'
    })
  }
})
```

- Jika sukses, pengguna akan tetap berada di halaman /profile, bila tidak, akan dilempar ke /login.



Sign in

Email Address *
narindal

Password *

☐ Remember me

SIGN IN

[Don't have an account? Sign Up](#)

- Buka halaman /pages/home.js. Di sini, kita akan melakukan verifikasi menggunakan token ke endpoint /verify sehingga hanya user yang memiliki token valid yang dapat mengakses halaman tersebut.

```
const home = () => {
  // State untuk mengecek apakah user sudah login atau belum
  const [isLogin, setIsLogin] = React.useState(false)

  // Modifikasi kode di bawah ini untuk mengambil data dari localStorage
  React.useEffect(() => {
    // 1. Ambil data 'user' dan 'token' dari localStorage
    const user = localStorage.getItem('user')
    const token = localStorage.getItem('token')

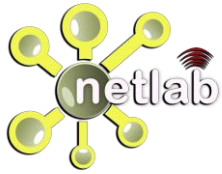
    // 2. Lempar ke halaman login bila user atau token tidak ada
    if (!user || !token) {
      window.location.href = '/login'
    }

    // 3. definisikan fungsi verifikasi token
    // fungsi ini akan melakukan HTTP POST request ke endpoint
    // /verify pada backend dengan mengirimkan token yang didapat dari localStorage
    // bila response status-nya 200 dan id dari response sama dengan id user pada
    // set isLogin menjadi true. bila tidak, redirect ke halaman login
    const verifyToken = async () => {
      await axios.post('http://localhost:3000/verify', {
        jwt: token
      }).then((res) => {
        if (res.status === 200 && res.data.id === JSON.parse(user).id) {

```

```
        // bila response status-nya 200 dan id dari response sama dengan id user pada localStorage,
        // set isLogin menjadi true. bila tidak, redirect ke halaman login
        const verifyToken = async () => {
          await axios.post('http://localhost:3000/verify', {
            jwt: token
          }).then((res) => {
            if (res.status === 200 && res.data.id === JSON.parse(user).id) {
              console.log('token valid')
              setIsLogin(true)
            } else {
              alert('Token Salah')
              window.location.href = '/login'
            }
          }).catch((err) => {
            console.log(err)
            alert('token kadaluarsa')
            window.location.href = '/login'
          })
        }

        // 4. Panggil fungsi verifikasi token
        verifyToken()
      }, [])
    }
  })
}
```



NETWORK LABORATORY

Electrical Engineering Department, 2nd floor
Universitas Indonesia
Depok. 16424

- Pastikan case berikut dapat berjalan: a. Registrasi user b. Login c. Melihat profile d. Melihat halaman home e. Melihat halaman 404 (mengakses halaman selain /login, /register, /profile, dan /saja)
- Modifikasi halaman 404 menjadi semenarik mungkin!