

Nama : NARINDA GENTA ROSASIA

NIM : 320200401019

## PRAKTIKUM WEB DEVELOPMENT

Minggu ke-9, Sesi ke-2

Case Study

Github : [https://github.com/narindagenta/Modul18\\_narinda](https://github.com/narindagenta/Modul18_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
```

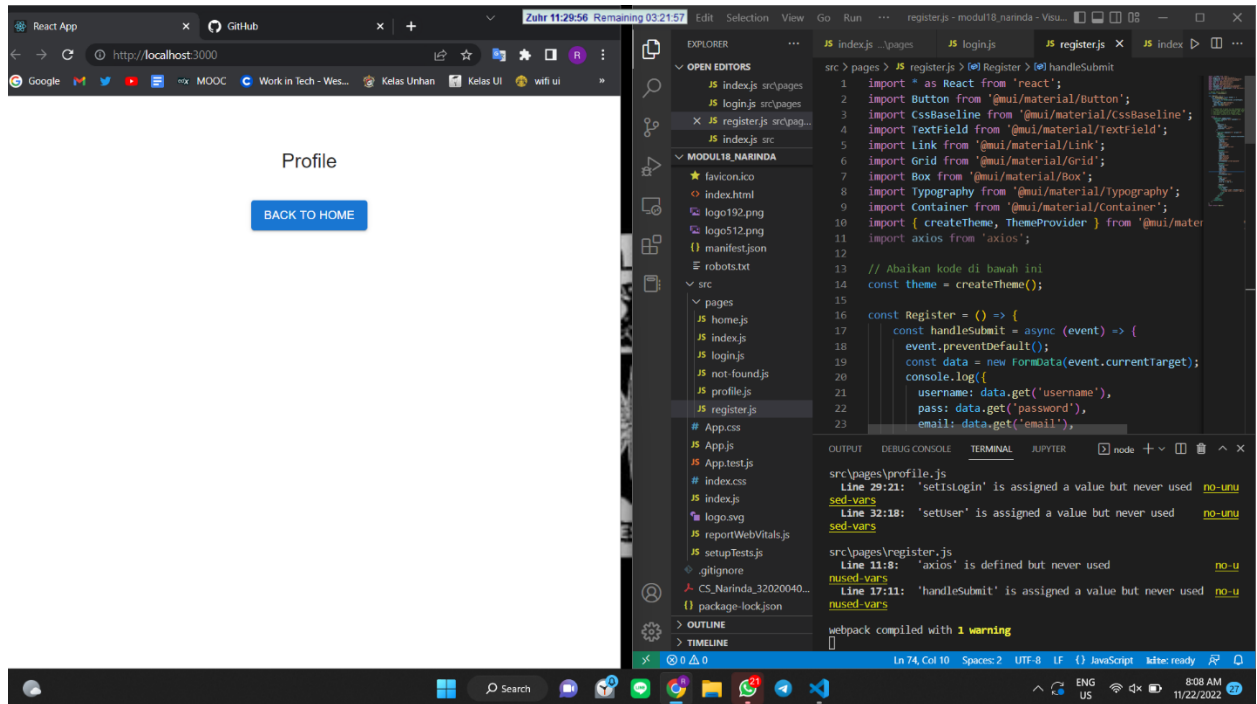
DEBUG CONSOLE    TERMINAL    JUPYTER



- 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 theme. At the top, there is a status bar with the text "Fajr: 03:48:35 Remaining 04:10:57". Below the address bar, there is a navigation bar with various icons and text links. The main content area is white and contains the following elements:

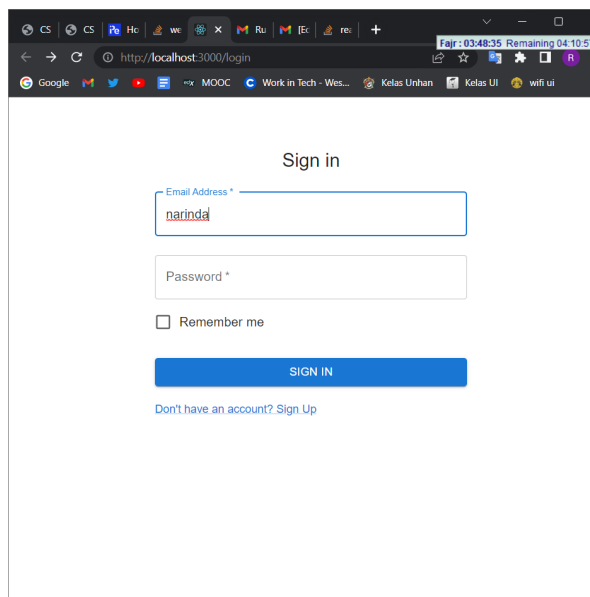
- The text "Sign in" centered at the top of the form.
- An input field labeled "Email Address \*" with the text "narinda" entered.
- An input field labeled "Password \*" which is currently empty.
- A checkbox labeled "Remember me" which is unchecked.
- A blue button labeled "SIGN IN".
- A link below the button that reads "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.





# NETWORK LABORATORY

Electrical Engineering Department, 2<sup>nd</sup> floor  
Universitas Indonesia  
Depok. 16424

The image shows a web browser and a code editor (VS Code) side-by-side. The browser displays a profile page at `http://localhost:3000/profile`. The page has a sidebar with 'Profile' and 'BACK TO HOME' buttons. The main content area shows a table with user information:

Key	Value
name	Narinda
pwd	nana
user	null

The code editor shows the `login.js` file in the `src/pages` directory. The code is as follows:

```
src > pages > JS login.js > Login > navigate
33 // jika berhasil, set localStorage 'user' dan 't
34 // jika gagal, tampilkan alert 'Login Gagal'
35 try {
36   const response = await axios.post('http://loca
37     email : data.get('email'),
38     password : data.get('password')
39   })
40   console.log(response)
41
42   localStorage.setItem('token', response.data.to
43   localStorage.setItem('id', response.data.id)
44   localStorage.setItem('username', response.data
45   localStorage.setItem('email', response.data.em
46
47   navigate('/profile')
48 } catch (err) {
49   alert('login gagal')
50 }
51
52 JS not-found.js
53 JS profile.js
54 JS register.js
55
56 App.css
57 App.js
58 App.test.js
59 index.css
60 index.js
61 logo.svg
62 reportWebVitals.js
63 setupTests.js
64 .gitignore
65 CS_Narinda_32020040...
66 package-lock.json
67
68 > OUTLINE
69 > TIMELINE
70
71 webpack compiled with 1 warning
```

The browser's developer tools show the 'Application' tab with the 'Storage' section expanded, showing the 'localStorage' item. The 'Console' tab shows the following messages:

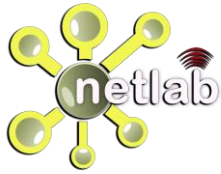
```
src > pages > JS login.js
Line 29:21: 'setIslogin' is assigned a value but never used
Line 32:18: 'setUser' is assigned a value but never used
src > pages > JS register.js
Line 11:8: 'axios' is defined but never used
Line 17:11: 'handleSubmit' is assigned a value but never used
```

The code editor also shows the `register.js` file in the `src/pages` directory. The code is as follows:

```
src > pages > JS register.js > Register > handleSubmit > response > email
26 // 1. Lakukan Axios POST ke API Register pada back
27 // body yang digunakan adalah username, email, dan
28 // jika berhasil, redirect ke halaman login
29 // jika gagal, tampilkan alert 'Register Gagal'
30 try {
31   const response = await axios.post('http://loca
32     username : data.get('username'),
33     email : data.get('email'),
34     password : data.get('password')
35   })
36   navigate('/login')
37 } catch (err) {
38   alert('tidak bisa daftar');
39 }
40
41 return (
42   <ThemeProvider theme={theme}>
43     <Container component="main" maxWidth="xs">
44       <CssBaseline />
45     </Container>
46   )
47
48 App.css
49 App.js
50 App.test.js
51 index.css
52 index.js
53 logo.svg
54 reportWebVitals.js
55 setupTests.js
56 .gitignore
57 CS_Narinda_32020040...
58 package-lock.json
59
60 > OUTLINE
61 > TIMELINE
62
63 webpack compiled with 1 warning
```

The browser's developer tools show the 'Application' tab with the 'Storage' section expanded, showing the 'localStorage' item. The 'Console' tab shows the following messages:

```
src > pages > JS register.js
Line 11:8: 'axios' is defined but never used
Line 17:11: 'handleSubmit' is assigned a value but never used
```



The image displays a web application running on a local server (localhost:3000). The application has two main pages: a registration page and a login page.

**Registration Page (http://localhost:3000/register):**

- Fields: Username, Password, Email.
- Buttons: REGISTER, LOGIN (link).
- Message: "Already have an account? Sign In".

**Login Page (http://localhost:3000/login):**

- Fields: Email Address, Password.
- Buttons: SIGN IN, SIGN UP (link).
- Message: "Don't have an account? Sign Up".

**Code Editor (VS Code):**

- File Explorer shows the project structure: src > pages > register.js, login.js, index.js.
- Register.js code (Line 31):

```
const response = await axios.post('http://localhost:3001/register', {
  username: data.get('username'),
  password: data.get('password'),
  email: data.get('email')
});
```
- Login.js code (Line 31):

```
const response = await axios.post('http://localhost:3001/login', {
  email: data.get('email'),
  password: data.get('password')
});
```

**Terminal:**

- Shows the command: `npm run dev`.
- Output: `webpack compiled with 1 warning`.

**System Tray:**

- Weather: 88°F Mostly sunny.
- Time: 9:52 AM 11/22/2022.

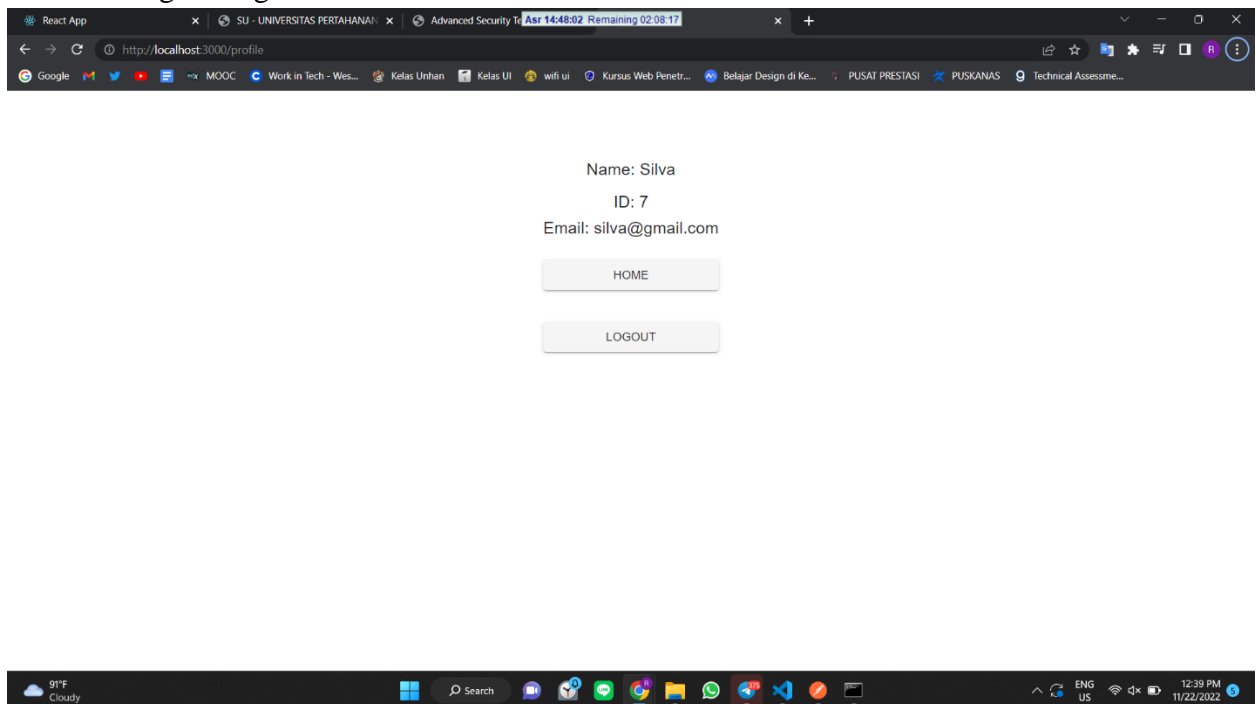


```
SQL Shell (psql)
Server [localhost]:
Database [postgres]:
Port [5432]:
Username [postgres]:
Password for user postgres:
psql (14.5)
WARNING: Console code page (437) differs from Windows code page (1252)
8-bit characters might not work correctly. See psql reference
page "Notes for Windows users" for details.
Type "help" for help.

postgres=# \c modul17
You are now connected to database "modul17" as user "postgres".
modul17=# select * from unhan_modul_17;
 id | username | email | password
-----+-----+-----+-----
  1 | narin | narind | $2b$10$1bAHMySiyIMM//.Qw.UHKOIj2uKRfmuA6j6qLCH2JbRLwCswNXG6T1
  3 | genta | genta@gmail.com | $2b$10$w.r9h2p66hi4nM9LRnGwbujY1fP8dhhBwX0RcZHPcVcrnAz5p8upy
  4 | rosa | rosa@gmail.com | $2b$10$RpEyFQtdfTu.C4Bu9oKqIOa2Uc3Hu9Do0Iur911c8LYG6dJK0tCWq
  5 | NARINDAGENTA | ngentasiacc@gmail.com | $2b$10$27VuvKBUu0JH5GbKdLokLOAirJsItfxHqBq2Va1d7t5pyripnEHN2
(4 rows)

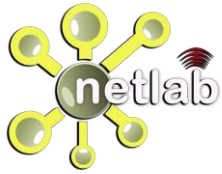
modul17=#
```

Mencoba login dengan user baru



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

```
// bila response status-nya 200 dan id dari response sama dengan id user pada localStorage,  
// set isLoggedIn 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()  
}, [])  
  
const handleToggle = () => {
```



## Halaman profile

The screenshot shows a web browser at `http://localhost:3000` displaying a profile page. The page content includes:

# Hello!

You can view this page because you're logged in.

[Go to your profile.](#)

My sticky footer can be found here.  
Copyright © Your Website 2022.

The background shows the VS Code editor with the following files open:

- `src > pages > home.js`
- `src > pages > login.js`
- `src > pages > profile.js`
- `src > pages > home.js`
- `src > pages > index.js`
- `src > pages > register.js`
- `src > pages > not-found.js`
- `src > pages > app.js`
- `src > pages > app.test.js`
- `src > pages > index.css`
- `src > pages > logo.svg`
- `src > pages > reportWebVitals.js`
- `src > pages > setupTests.js`
- `src > pages > .gitignore`
- `src > pages > CS_Narinda_32020040...`
- `src > pages > package-lock.json`
- `src > pages > package.json`
- `src > pages > README.md`
- `src > pages > OUTLINE`
- `src > pages > TIMELINE`

The code in `home.js` is as follows:

```
// verify pada backend dengan mengirimkan tok
// bila response status-nya 200 dan id dari re
// set islogin menjadi true, bila tidak, redir

const verify = async() =>{
  try {
    const response = await axios.post('http:
    token: localStorage.getItem('token')
  )
  if(response.status == 200){
    setIslogin(true)
  }else{
    navigate('/login')
  }
} catch (error) {
  navigate('/login')
}

// 4. Panggil fungsi verifikasi token
verify()
```

The terminal output shows the following warnings:

- Line 38:13: 'id' is assigned a value but never used `no-unused-`
- Line 42:12: Expected a conditional expression and instead saw an as `no-cond-as`
- Line 57:34: Expected '===' and instead saw '==' `eqeqeq`
- Line 71:8: React Hook React.useEffect has a missing dependency: 'n `react-hook`

The status bar at the bottom indicates: `Ln 129, Col 24 Spaces: 4 UTF-8 LF JavaScript kites: ready`



## Keseluruhan web

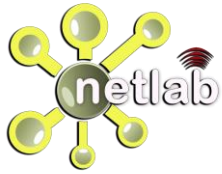
- Pastikan case berikut dapat berjalan:
  - a. Registrasi user

A screenshot of a web browser window displaying a registration form. The browser's address bar shows the URL "http://localhost:3000/register". The form is titled "Register" and contains three input fields: "Username \*", "Password \*", and "Email \*". Below these fields is a blue button labeled "REGISTER". At the bottom of the form, there is a link that says "Already have an account? Sign In". The browser's taskbar at the bottom shows various application icons and the system clock indicating 1:41 PM on 11/22/2022.

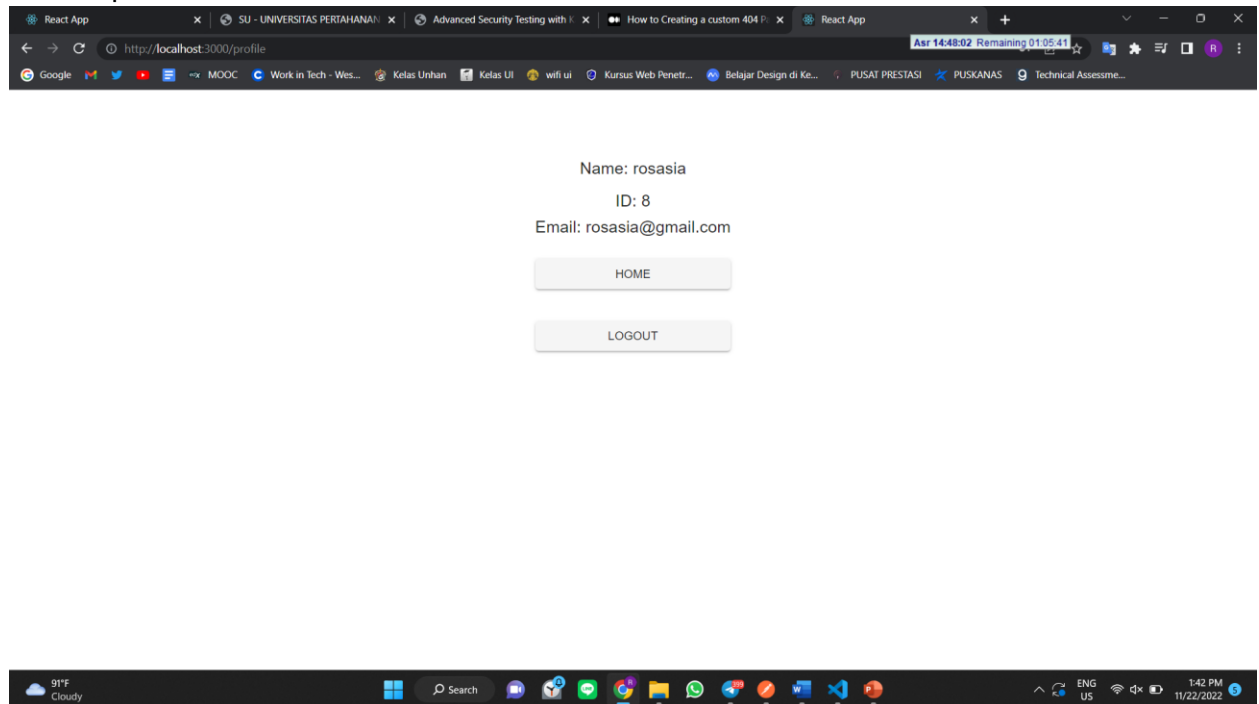


b. Login

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 background with a "Sign in" heading. Below the heading are two input fields: "Email Address \*" with the value "silva@gmail.com" and "Password \*" with masked characters "\*\*\*\*\*". There is a "Remember me" checkbox below the password field. A blue "SIGN IN" button is positioned below the inputs. At the bottom of the form, there is a link that says "Don't have an account? Sign Up". The browser's taskbar at the bottom shows various application icons and the system clock indicating 1:42 PM on 11/22/2022.

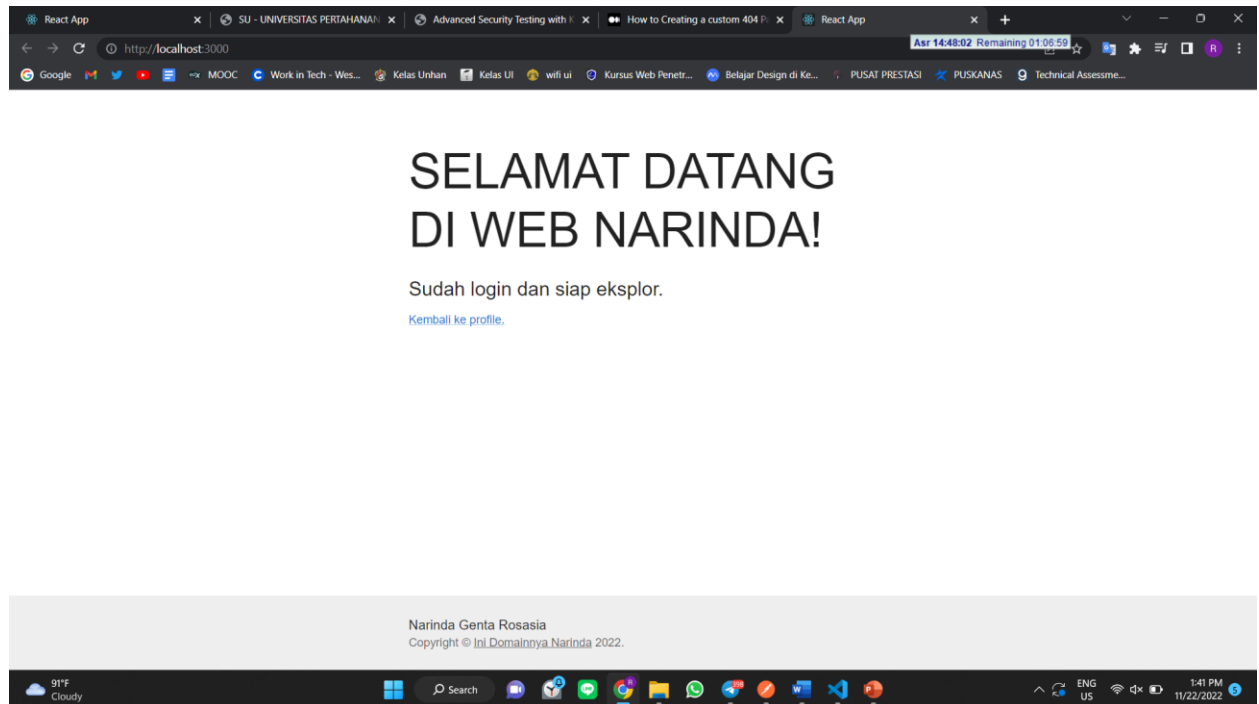


c. Melihat profile

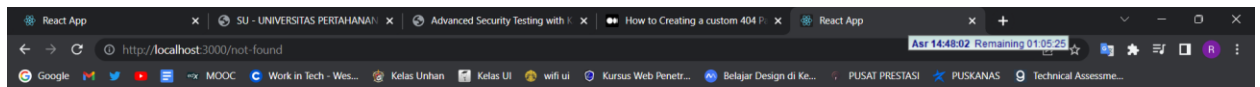




d. Melihat halaman home



e. Melihat halaman 404 (mengakses halaman selain /login, /register, /profile, dan / saja)



**Halaman tidak ditemukan**

