

Laporan Praktikum
Mata Kuliah Pemrograman Web



Pertemuan 6. Praktikum 6
“SESSION”

Dosen Pengampu : Willdan Aprizal Aripin, S.Pd., M.Kom.

Disusun Oleh :

Nuril Khairiyah

2300231

PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA
2024

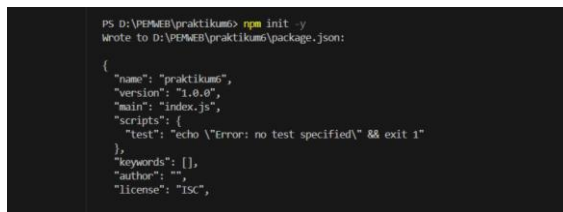
I. PENDAHULUAN

Session adalah mekanisme yang digunakan dalam aplikasi web untuk menyimpan data pengguna secara sementara selama mereka berinteraksi dengan aplikasi. Fungsinya adalah untuk mempertahankan informasi pengguna di antara permintaan HTTP yang berbeda, memungkinkan aplikasi untuk "mengingat" pengguna selama sesi berlangsung. Dalam Node.js, session digunakan untuk melacak dan menyimpan status pengguna agar data seperti informasi login, preferensi atau pengaturan tertentu tetap tersedia dan konsisten sepanjang penggunaan aplikasi tersebut meskipun terjadi beberapa permintaan atau navigasi halaman.

II. ALAT DAN BAHAN

- Visual Studio Code
- Chrome
- Laptop
- Node.js
- Xampp
- MySQL

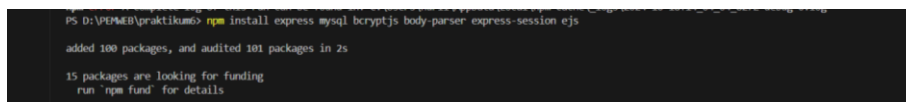
III. PENJELASAN



```
PS D:\PEMWEB\praktikum> npm init -y
Wrote to D:\PEMWEB\praktikum\package.json:

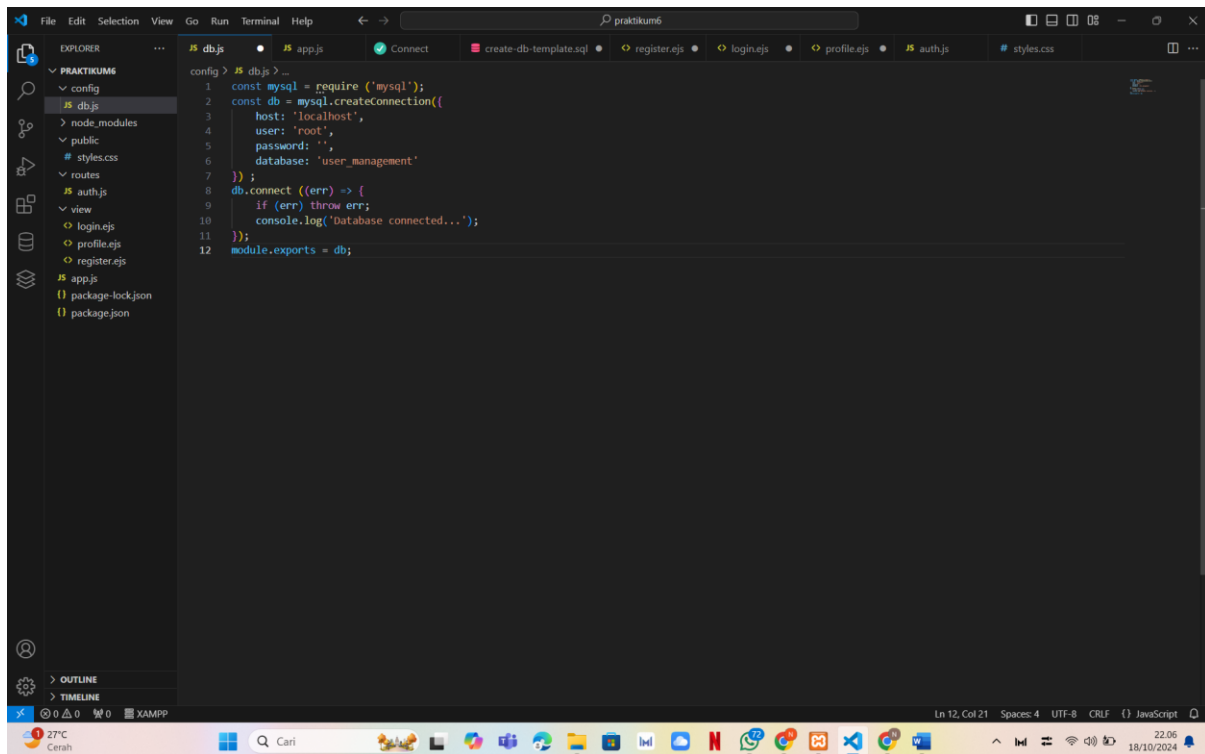
{
  "name": "praktikum",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
}
```

Perintah `npm init -y` digunakan untuk menginisialisasi proyek Node.js dengan membuat file `package.json` secara otomatis.



```
PS D:\PEMWEB\praktikum> npm install express mysql bcryptjs body-parser express-session ejs
added 100 packages, and audited 101 packages in 2s
15 packages are looking for funding
run npm fund for details
```

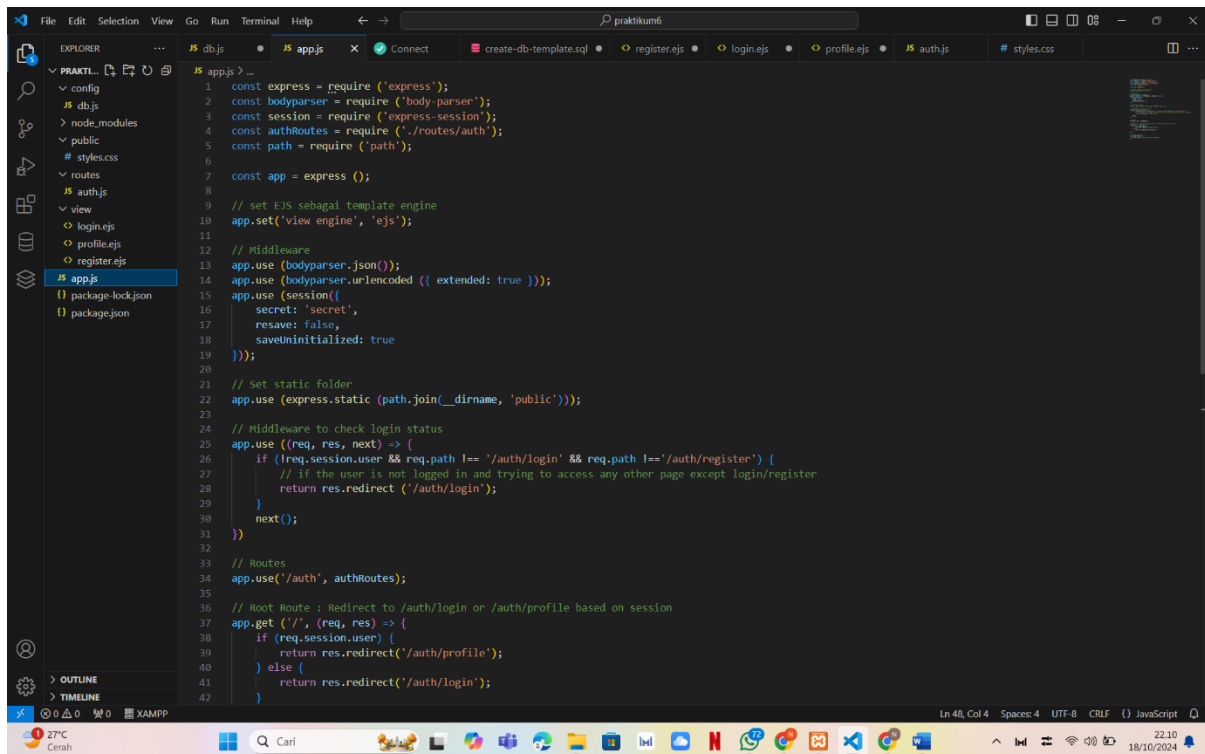
Perintah ini digunakan untuk menginstal beberapa package atau modul Node.js yang diperlukan dalam sebuah proyek. Yang di install yaitu `express`, `mysql`, `bcryptjs`, `body-parser` dan `express-session`



```
1 const mysql = require('mysql');
2 const db = mysql.createConnection({
3   host: 'localhost',
4   user: 'root',
5   password: '',
6   database: 'user_management'
7 });
8 db.connect((err) => {
9   if (err) throw err;
10  console.log('Database connected...');
11 });
12 module.exports = db;
```

Kode pada file db.js digunakan untuk menghubungkan Node.js ke database MySQL. Pada bagian awal modul mysql diimpor ke dalam aplikasi. Modul ini memungkinkan tersambung dengan database MySQL. Selanjutnya membuat koneksi ke database menggunakan `mysql.createConnection()`. Di sini beberapa parameter penting diberikan, seperti alamat host (`localhost`), nama pengguna (`root`) dan nama database (`user_management`). Selain itu ada juga bagian password yang diisi dengan string kosong menandakan bahwa saat ini tidak ada password yang digunakan untuk mengakses database.

Setelah mendefinisikan koneksi aplikasi mencoba untuk terhubung ke database melalui fungsi `db.connect()`. Jika koneksi gagal maka akan melemparkan error dan berhenti. Namun jika koneksi berhasil pesan "Database connected..." akan ditampilkan di terminal sebagai tanda bahwa koneksi ke MySQL berhasil dibuat. Terakhir objek db yang merupakan koneksi ke database diekspor menggunakan module Exports. Hal ini memungkinkan objek db digunakan di file lain dalam aplikasi, misalnya untuk menjalankan kueri SQL atau operasi lain yang berhubungan dengan database.



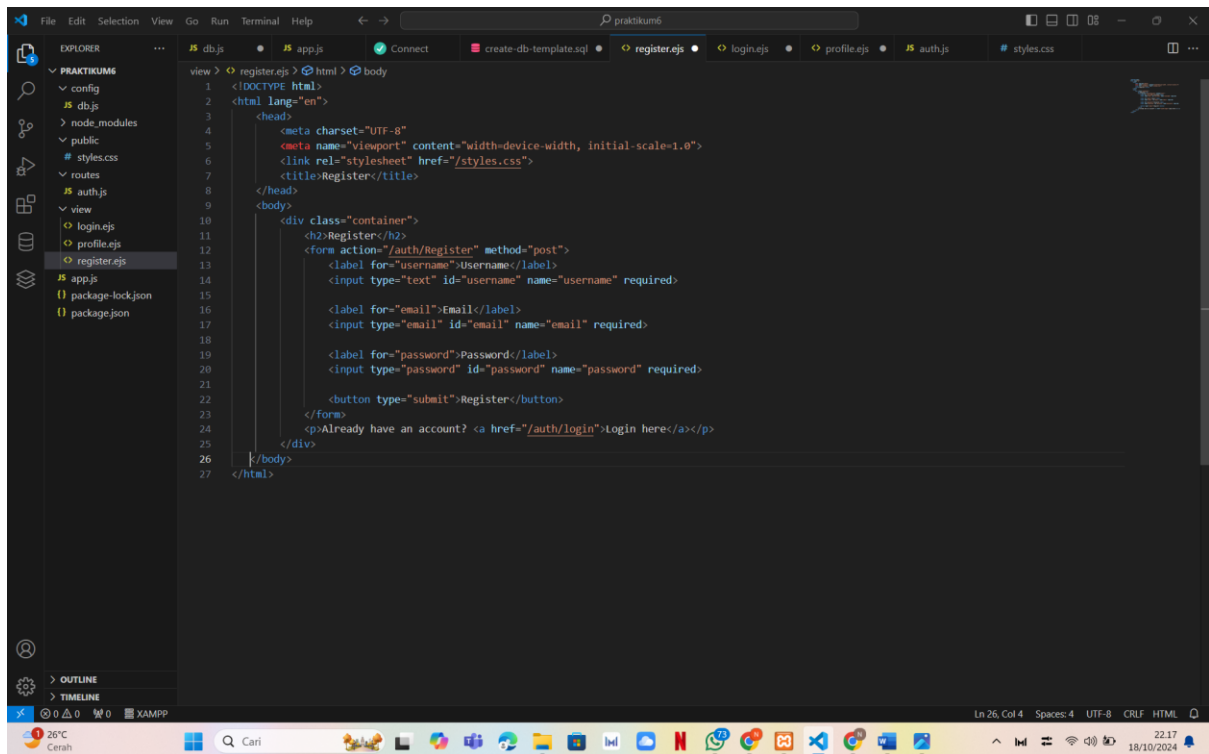
```
1 const express = require('express');
2 const bodyParser = require('body-parser');
3 const session = require('express-session');
4 const authRoutes = require('./routes/auth');
5 const path = require('path');
6
7 const app = express();
8
9 // set EJS sebagai template engine
10 app.set('view engine', 'ejs');
11
12 // Middleware
13 app.use(bodyParser.json());
14 app.use(bodyParser.urlencoded({ extended: true }));
15 app.use(session({
16   secret: 'secret',
17   resave: false,
18   saveUninitialized: true
19 }));
20
21 // Set static folder
22 app.use(express.static(path.join(__dirname, 'public')));
23
24 // Middleware to check login status
25 app.use((req, res, next) => {
26   if (!req.session.user && req.path !== '/auth/login' && req.path !== '/auth/register') {
27     // If the user is not logged in and trying to access any other page except login/register
28     return res.redirect('/auth/login');
29   }
30   next();
31 })
32
33 // Routes
34 app.use('/auth', authRoutes);
35
36 // Root Route : Redirect to /auth/login or /auth/profile based on session
37 app.get('/', (req, res) => {
38   if (req.session.user) {
39     return res.redirect('/auth/profile');
40   } else {
41     return res.redirect('/auth/login');
42   }
43 })
```

Kode ini merupakan bagian dari Node.js yang menggunakan framework Express untuk mengatur server dan merespons permintaan HTTP. Beberapa modul atau paket yang diperlukan diimpor, termasuk `express`, `body-parser`, `express-session`, dan `path`. Modul-modul ini memberikan berbagai fungsi yang dibutuhkan untuk mengelola server memproses data yang diterima mengelola sesi pengguna serta menangani rute aplikasi.

Folder statis untuk menyimpan aset seperti file CSS, gambar, dan JavaScript diatur menggunakan `express.static()`. Ini memastikan bahwa file di dalam folder `public` bisa diakses langsung oleh klien. Selain itu, ada middleware khusus yang memeriksa apakah pengguna sudah login atau belum. Jika pengguna belum login dan mencoba mengakses halaman selain halaman login atau register pengguna akan diarahkan kembali ke halaman login.

Halaman kemudian mengatur beberapa rute. Rute `'/auth'` menangani semua permintaan yang berkaitan dengan autentikasi, seperti login, register, dan logout. Rute utama juga diatur, di mana aplikasi akan mengarahkan pengguna ke halaman profil jika mereka sudah login atau ke halaman login jika belum.

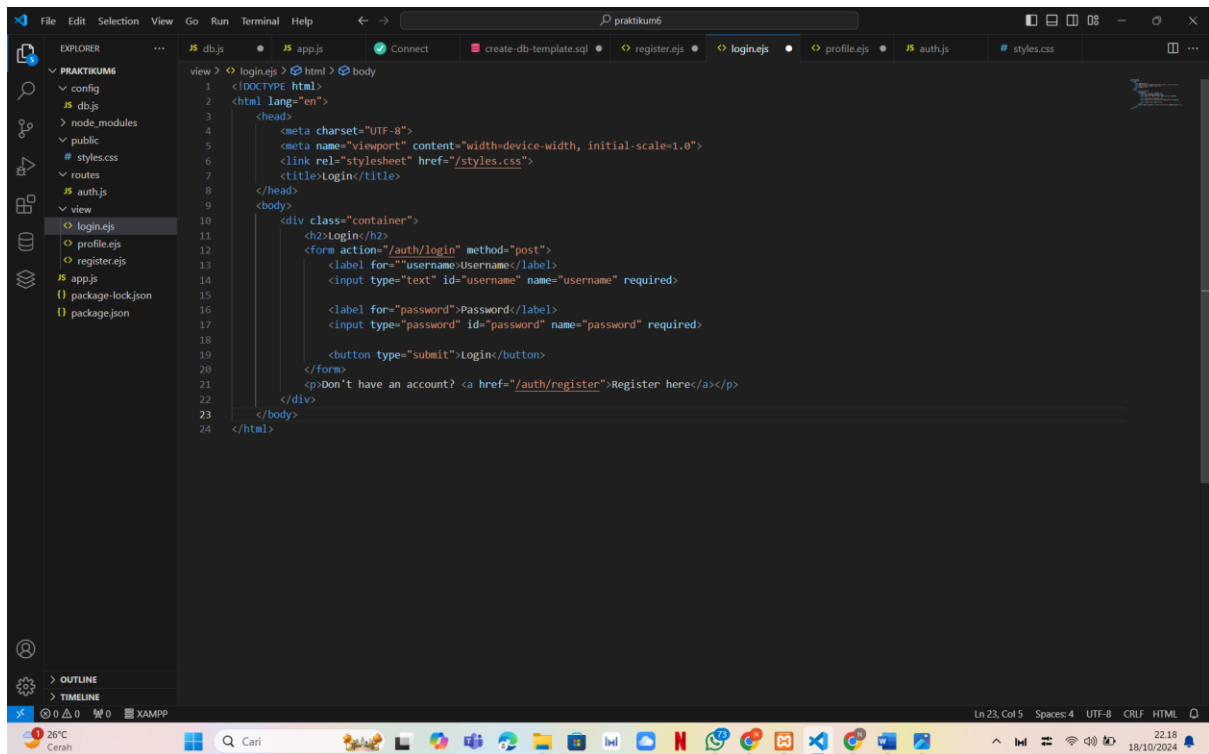
Terakhir, server dijalankan pada port 3000. Ketika server mulai berjalan, pesan "server running on port 3000" akan ditampilkan di terminal untuk menunjukkan bahwa server siap menerima permintaan.



```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <link rel="stylesheet" href="/styles.css">
7     <title>Register</title>
8   </head>
9   <body>
10    <div class="container">
11      <h2>Register</h2>
12      <form action="/auth/register" method="post">
13        <label for="username">Username</label>
14        <input type="text" id="username" name="username" required>
15
16        <label for="email">Email</label>
17        <input type="email" id="email" name="email" required>
18
19        <label for="password">Password</label>
20        <input type="password" id="password" name="password" required>
21
22        <button type="submit">Register</button>
23      </form>
24      <p>Already have an account? <a href="/auth/login">Login here</a></p>
25    </div>
26  </body>
27 </html>
```

Kode HTML tersebut adalah halaman pendaftaran (register) yang memungkinkan pengguna untuk mendaftarkan akun baru. Bagian `<head>` mendefinisikan metadata halaman, termasuk pengaturan karakter encoding menggunakan `<meta charset="UTF-8">` yang memastikan teks tampil dengan benar dan pengaturan agar tampilan halaman responsif dengan `<meta name="viewport" content="width=device-width, initial-scale=1.0">`. Tautan ke file CSS eksternal `/styles.css` ditambahkan untuk menerapkan gaya pada halaman dan judul halaman diatur menjadi "Register" melalui elemen `<title>`.

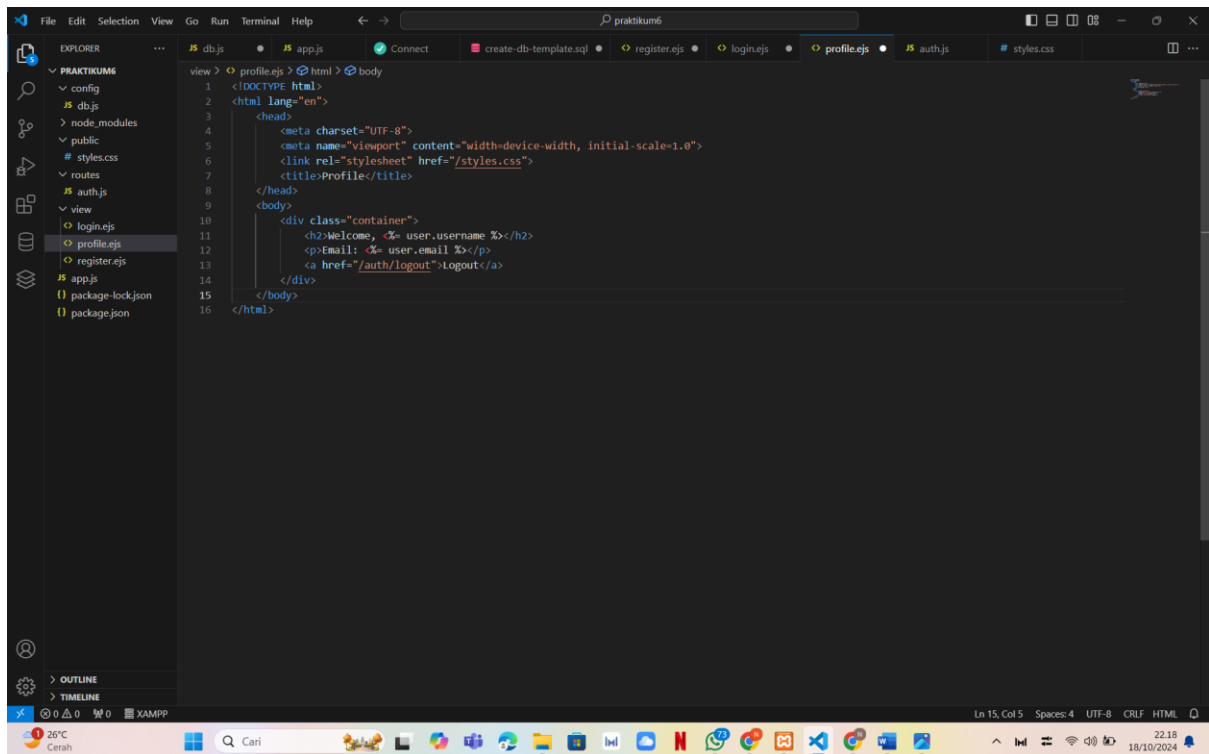
Di dalam `<body>`, ada sebuah `<div>` dengan kelas "container" yang membungkus seluruh konten halaman. Halaman ini menampilkan heading `<h2>` dengan teks "Register" dan sebuah form untuk mengumpulkan data pengguna, seperti username, email, dan password. Formulir tersebut menggunakan metode "POST" untuk mengirim data ke server melalui URL `/auth/register` ketika tombol "Register" ditekan. Setiap input dilengkapi dengan atribut `required` untuk memastikan bahwa pengguna wajib mengisinya sebelum mengirim form. Di bawah formulir terdapat teks yang menyarankan pengguna yang sudah memiliki akun untuk login melalui tautan ke halaman `/auth/login`.



```
1 <!DOCTYPE html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8">
5     <meta name="viewport" content="width=device-width, initial-scale=1.0">
6     <link rel="stylesheet" href="/styles.css">
7     <title>Login</title>
8   </head>
9   <body>
10    <div class="container">
11      <h2>Login</h2>
12      <form action="/auth/login" method="post">
13        <label for="username">Username</label>
14        <input type="text" id="username" name="username" required>
15
16        <label for="password">Password</label>
17        <input type="password" id="password" name="password" required>
18
19        <button type="submit">Login</button>
20      </form>
21      <p>don't have an account? <a href="/auth/register">Register here</a></p>
22    </div>
23  </body>
24 </html>
```

Kode HTML tersebut adalah halaman login yang dirancang untuk memungkinkan pengguna masuk ke dalam aplikasi. Pada bagian `<head>` ditentukan karakter encoding dengan `<meta charset="UTF-8">` untuk memastikan teks ditampilkan dengan benar, serta pengaturan tampilan responsif melalui `<meta name="viewport" content="width=device-width, initial-scale=1.0">`. Ada juga tautan ke file CSS eksternal (`/styles.css`) yang digunakan untuk menerapkan gaya pada halaman, dan judul halaman ditetapkan menjadi "Login".

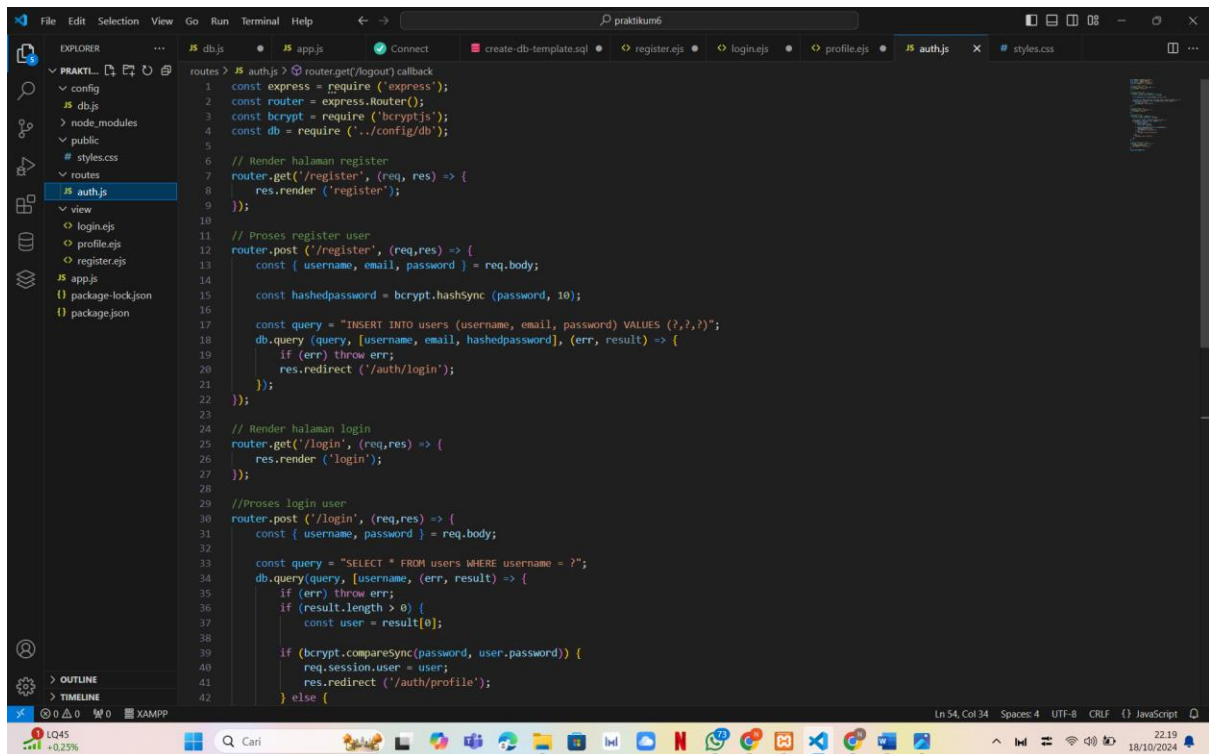
Di dalam `<body>`, ada sebuah `<div>` dengan kelas "container" yang membungkus seluruh konten utama halaman. Halaman ini menampilkan heading `<h2>` dengan teks "Login" dan sebuah form yang meminta pengguna untuk mengisi username dan password. Formulir tersebut menggunakan metode "POST" untuk mengirim data ke server melalui URL `/auth/login` saat tombol "Login" ditekan. Setiap input memiliki atribut `required` yang mengharuskan pengguna mengisi semua data sebelum dapat mengirim form. Di bawah formulir, terdapat sebuah paragraf yang memberikan pilihan kepada pengguna yang belum memiliki akun untuk mendaftar dengan mengklik tautan ke halaman `/auth/register`.

A screenshot of a code editor window with a dark theme. The Explorer sidebar on the left shows a project structure with folders like 'config', 'node_modules', 'public', 'styles.css', 'routes', 'auth.js', 'view', 'login.ejs', 'profile.ejs', 'register.ejs', 'app.js', and files like 'package-lock.json' and 'package.json'. The 'profile.ejs' file is selected and its content is displayed in the main editor. The code is an HTML template for a user profile page. It includes a DOCTYPE declaration, a language attribute, a UTF-8 charset meta tag, a viewport meta tag for responsiveness, a link to an external CSS file, and a title 'Profile'. The body contains a container div with a welcome message, the user's username, their email, and a link to the logout page.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <link rel="stylesheet" href="/styles.css">
7   <title>Profile</title>
8 </head>
9 <body>
10   <div class="container">
11     <h2>Welcome, <%= user.username %></h2>
12     <p>Email: <%= user.email %></p>
13     <a href="/auth/logout">Logout</a>
14   </div>
15 </body>
16 </html>
```

Kode HTML tersebut adalah halaman profil yang menampilkan informasi pengguna yang sedang login. Bagian `<head>` mengatur karakter encoding dengan `<meta charset="UTF-8">` untuk memastikan tampilan teks yang benar, serta menggunakan `<meta name="viewport" content="width=device-width, initial-scale=1.0">` agar halaman dapat ditampilkan secara responsif. Halaman ini juga terhubung dengan file CSS eksternal (`'/styles.css'`) yang digunakan untuk menambahkan gaya pada elemen-elemen di dalamnya dan judul halaman ditetapkan sebagai "Profile".

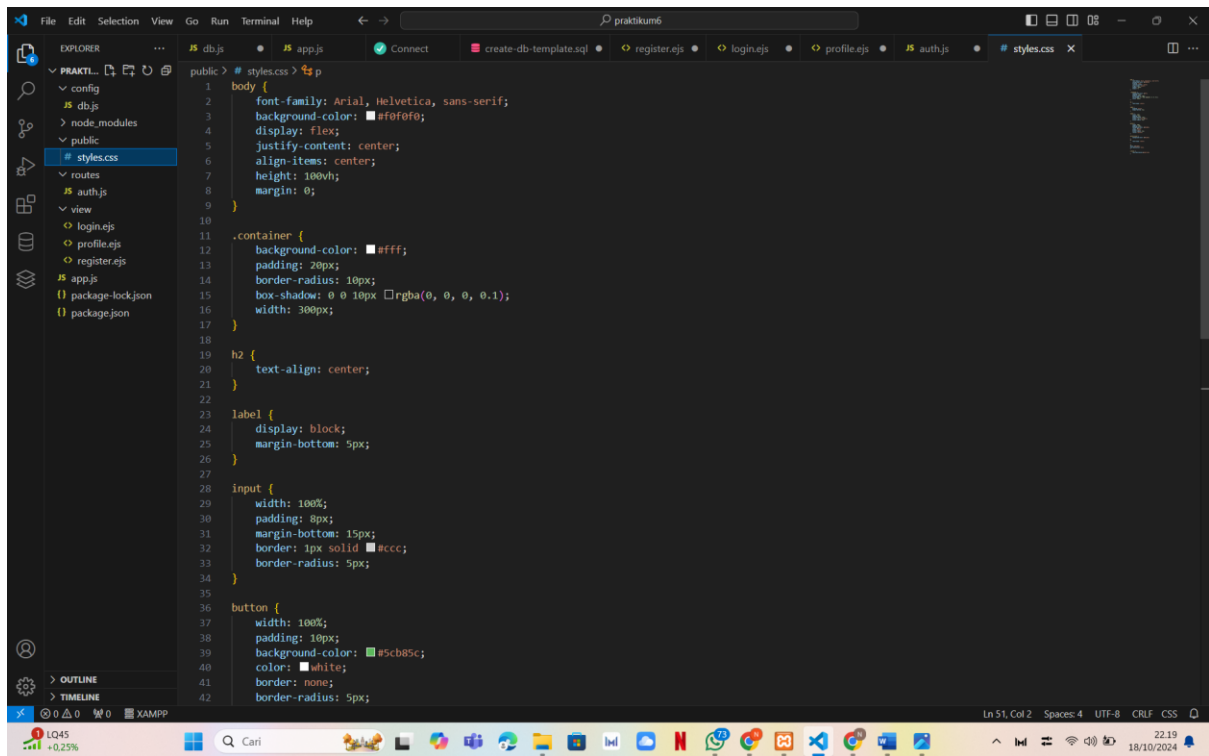
Di bagian `<body>`, terdapat sebuah `<div>` dengan kelas "container" yang berfungsi untuk membungkus konten utama. Halaman ini menyambut pengguna dengan pesan selamat datang yang memuat nama pengguna, yang ditampilkan menggunakan sintaks `<%= user.username %>` di mana data ini berasal dari server (melalui template engine seperti EJS). Selain itu, email pengguna juga ditampilkan dengan cara yang sama, yaitu `<%= user.email %>`. Di bagian bawah, terdapat tautan untuk melakukan logout dari akun, yang mengarah ke `"/auth/logout"`.



```
1 const express = require('express');
2 const router = express.Router();
3 const bcrypt = require('bcryptjs');
4 const db = require('../config/db');
5
6 // Render halaman register
7 router.get('/register', (req, res) => {
8   res.render('register');
9 });
10
11 // Proses register user
12 router.post('/register', (req, res) => {
13   const { username, email, password } = req.body;
14
15   const hashedpassword = bcrypt.hashSync(password, 10);
16
17   const query = "INSERT INTO users (username, email, password) VALUES (?, ?, ?)";
18   db.query(query, [username, email, hashedpassword], (err, result) => {
19     if (err) throw err;
20     res.redirect('/auth/login');
21   });
22 });
23
24 // Render halaman login
25 router.get('/login', (req, res) => {
26   res.render('login');
27 });
28
29 // Proses login user
30 router.post('/login', (req, res) => {
31   const { username, password } = req.body;
32
33   const query = "SELECT * FROM users WHERE username = ?";
34   db.query(query, [username], (err, result) => {
35     if (err) throw err;
36     if (result.length > 0) {
37       const user = result[0];
38
39       if (bcrypt.compareSync(password, user.password)) {
40         req.session.user = user;
41         res.redirect('/auth/profile');
42       } else {
43         // Password salah
44       }
45     }
46   });
47 });
```

Kode JavaScript ini menggunakan framework Express untuk menangani rute autentikasi pengguna dalam aplikasi web. Pertama Express diimpor dan router dibuat untuk mengelola rute sedangkan bcryptjs digunakan untuk enkripsi kata sandi. Database dikonfigurasi melalui modul db. Rute pertama adalah GET /register yang menampilkan halaman pendaftaran (register.ejs). Rute kedua, POST /register, menangani proses pendaftaran pengguna. Data yang diterima dari form, yaitu username, email, dan password, dienkripsi menggunakan bcrypt dan disimpan ke dalam database. Jika proses pendaftaran berhasil, pengguna akan diarahkan ke halaman login.

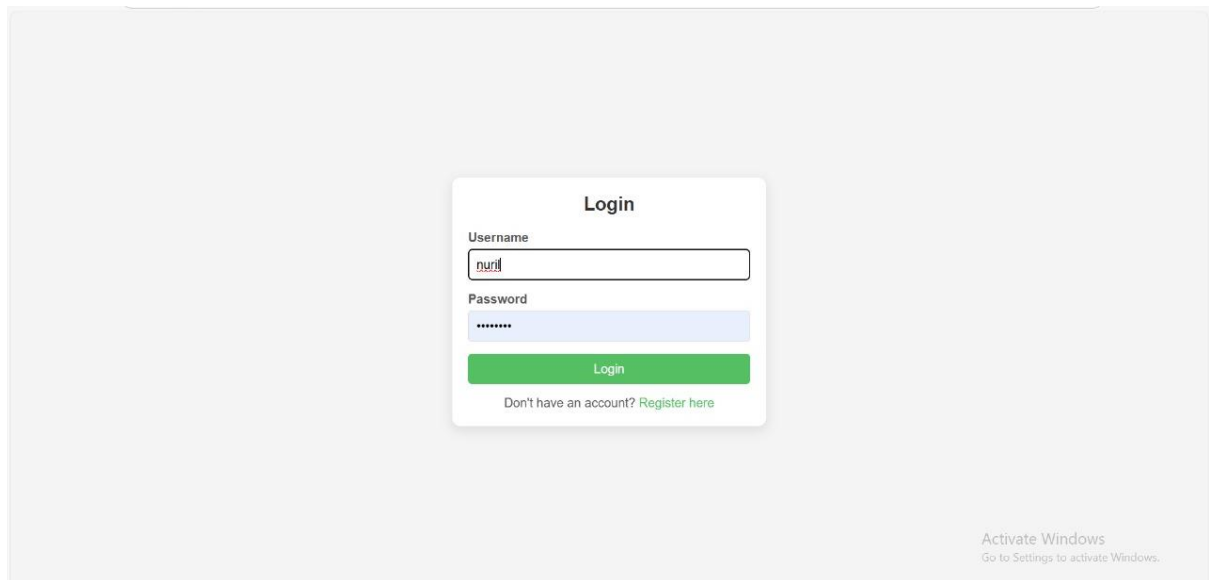
Rute GET /login menampilkan halaman login (login.ejs). Rute POST /login digunakan untuk memproses autentikasi pengguna. Setelah menerima username dan password, sistem memverifikasi apakah data pengguna ada di database. Jika ada bcrypt digunakan untuk memeriksa kecocokan antara kata sandi yang diinput dengan yang tersimpan di database. Jika berhasil sesi pengguna disimpan dan diarahkan ke halaman profil. Jika tidak pesan kesalahan akan ditampilkan. Ada juga rute untuk logout (GET /logout) yang mengakhiri sesi pengguna dan mengarahkan kembali ke halaman login.



```
1 body {
2   font-family: Arial, Helvetica, sans-serif;
3   background-color: #f0f0f0;
4   display: flex;
5   justify-content: center;
6   align-items: center;
7   height: 100vh;
8   margin: 0;
9 }
10
11 .container {
12   background-color: #fff;
13   padding: 20px;
14   border-radius: 10px;
15   box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
16   width: 300px;
17 }
18
19 h2 {
20   text-align: center;
21 }
22
23 label {
24   display: block;
25   margin-bottom: 5px;
26 }
27
28 input {
29   width: 100%;
30   padding: 8px;
31   margin-bottom: 15px;
32   border: 1px solid #ccc;
33   border-radius: 5px;
34 }
35
36 button {
37   width: 100%;
38   padding: 10px;
39   background-color: #5cb85c;
40   color: white;
41   border: none;
42   border-radius: 5px;
```

Berikut adalah style css yang digunakan pada halaman

HASIL



Register

Username

nuril

Email

nuril@upi.edu

Password

Register

Already have an account? [Login here](#)

Activate Windows
Go to Settings to activate Windows.

Welcome, nuril

Email: nuril@upi.edu

[Logout](#)

Activate Windows
Go to Settings to activate Windows.