

Laporan Praktikum
Mata Kuliah Pemrograman Berorientasi Objek



Pertemuan 7.Praktikum7

“Session Admin Perpustakaan menggunakan node js dan database

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

Disusun Oleh :
Siti Nurkholizah 2308807
3A - Sistem Informasi Kelautan

PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA

2024

I. PENDAHULUAN

Dalam pengembangan aplikasi web, autentikasi dan otorisasi merupakan aspek penting untuk memastikan hanya pengguna yang sah dapat mengakses fitur tertentu. Salah satu metode yang digunakan untuk melacak status login pengguna adalah session.

Session merupakan mekanisme di mana server menyimpan data sementara tentang pengguna selama interaksi dengan aplikasi. Ketika pengguna login, server membuat session unik yang menghubungkan pengguna dengan server melalui cookie. Dengan session, server dapat "mengingat" pengguna tanpa menyimpan informasi sensitif di sisi client, menjadikannya lebih aman.

II. TUJUAN

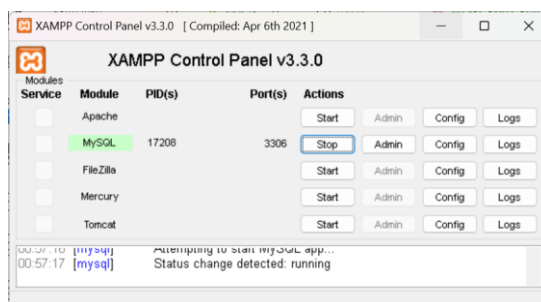
Dalam praktikum ini dapat mempelajari dan mengimplementasikan sistem autentikasi pengguna berbasis session menggunakan Node.js dan MySQL. Praktikum ini juga bertujuan untuk memahami cara mengenkripsi password menggunakan bcryptjs, serta mengelola session secara aman dengan express-session. Selain itu, penggunaan ejs sebagai template engine diharapkan dapat memberikan pengalaman dalam merender halaman web dinamis sesuai dengan status login pengguna. Dengan demikian, peserta dapat memahami konsep session, enkripsi password, serta pengelolaan autentikasi dan otorisasi dalam aplikasi web.

III. ALAT DAN BAHAN

- Laptop
- Node.js yang sudah terinstall di komputer.
- Visual Studio Code
- Browser web (Google Chrome)
- Terminal atau Command Prompt.
- File JS, EJS, dan CSS.

IV. LANGKAH KERJA

1. Sebelum koneksi database ke server aktifkan terlebih dahulu aplikasi XAMPP centang mysql sampai muncul port 3306



2. Buat database terlebih dahulu pada server xampp “user_management”

source code

```
CREATE DATABASE library ;
USE library;
CREATE TABLE admins (
  id INT AUTO_INCREMENT PRIMARY KEY,
  username VARCHAR(100) NOT NULL,
  email VARCHAR(200) NOT NULL,
  password VARCHAR(255) NOT NULL
);
```

Lalu run hingga muncul affectrow maka db otomatis telah dibuat

3. Memanggil npm init -y dalam terminal agar node js bisa dijalankan atau node_modules muncul dalam folder explorer
4. Menginstall npm express, mysql, mysql bcryptjs body-parser express-session dan ejs dalam terminal fungsinya agar package.json dan package-lock.json muncul pada folder explorer

Fungsi

- express: framework web untuk Node.js.
- mysql: driver untuk koneksi ke MySQL.
- bcryptjs: untuk hashing password.
- body-parser: untuk parsing request body.
- express-session: untuk manajemen sesi.
- ejs : untuk template engine.

5. Setelah itu buat folder dengan format struktur seperti ini

```
library-system/
|
├─ app.js
├─ package.json
├─ config/
|   └─ db.js
├─ public/
|   └─ style.css
├─ routes/
|   └─ auth.js
├─ views/
|   ├── login.ejs
|   ├── register.ejs
|   └─ profile.ejs
```

6. Input kodingan login.ejs, register.ejs, dan profile.ejs

Source code login.ejs dengan cssnya

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Login Admin Perpustakaan</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      margin: 0;
      padding: 0;
      display: flex;
      justify-content: center;
      align-items: center;
      height: 100vh;
      background-color: #b3f0df;
    }

    h2 {
      text-align: center;
    }

    form {
      display: flex;
      flex-direction: column;
      width: 300px;
      margin: 0 auto;
    }

    input {
      margin: 5px 0;
      padding: 10px;
      font-size: 16px;
    }

    button {
      padding: 10px;
      background-color: #108243;
      color: white;
      border: none;
      cursor: pointer;
    }
```

```

    button:hover {
        background-color: #77f19a;
    }

    label {
        font-weight: bold;
        margin-top: 10px;
    }

    .container {
        background-color: rgb(33, 213, 114);
        padding: 20px;
        border-radius: 8px;
        box-shadow: 0px 0px 10px rgb(8, 169, 126);
    }

    p {
        text-align: center;
    }

    a {
        color: #ffffff;
        text-decoration: none;
    }

    a:hover {
        text-decoration: underline;
    }
</style>
</head>
<body>
    <div class="container">
        <h2>Login Perpustakaan</h2>
        <form action="/auth/login" method="POST">
            <label for="username">Nama Admin</label>
            <input type="text" id="username" name="username"
placeholder="Masukkan nama admin" required>

            <label for="email">Email</label>
            <input type="email" id="email" name="email"
placeholder="Masukkan email admin" required>

```

```

        <label for="password">Kata Sandi</label>
        <input type="password" id="password" name="password"
placeholder="Masukkan kata sandi" required>

        <button type="submit">Login</button>
    </form>

    <p>Belum memiliki akun? <a href="/auth/register">Daftar
disini!</a></p>
</div>
</body>
</html>

```

Source code register.ejs dengan cssnya

```

<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <title>Registrasi Admin Perpustakaan</title>
    <style>
        body {
            font-family: Arial, sans-serif;
            margin: 0;
            padding: 0;
            display: flex;
            justify-content: center;
            align-items: center;
            height: 100vh;
            background-color: #b3f0df;
        }

        h2 {
            text-align: center;
        }
    </style>

```

```
form {  
  display: flex;  
  flex-direction: column;  
  width: 300px;  
  margin: 0 auto;  
}
```

```
input {  
  margin: 5px 0;  
  padding: 10px;  
  font-size: 16px;  
}
```

```
button {  
  padding: 10px;  
  background-color: #108243;  
  color: white;  
  border: none;  
  cursor: pointer;  
}
```

```
button:hover {  
  background-color: #77f19a;  
}
```

```
label {  
  font-weight: bold;  
  margin-top: 10px;  
}
```

```
.container {  
  background-color: rgb(33, 213, 114);  
  padding: 20px;  
  border-radius: 8px;  
  box-shadow: 0px 0px 10px rgb(8, 169, 126);  
}
```

```
p {  
  text-align: center;  
}
```

```

a {
    color: #ffffff;
    text-decoration: none;
}

a:hover {
    text-decoration: underline;
}
</style>
</head>
<body>
<div class="container">
    <h2>Register Perpustakaan</h2>
    <form action="/auth/register" method="POST">
        <label for="username">Nama Admin</label>
        <input type="text" id="username" name="username"
placeholder="Masukkan nama admin" required><br>

        <label for="email">Email</label>
        <input type="email" id="email" name="email"
placeholder="Masukkan email admin" required><br>

        <label for="password">Kata Sandi</label>
        <input type="password" id="password" name="password"
placeholder="Masukkan kata sandi" required><br>

        <button type="submit">Login</button>
    </form>
    <p>Sudah memiliki akun? <a href="/auth/login">Login
disini!</a></p>
    </div>
</body>
</body>
</html>

```


Source code profile.ejs dengan cssnya

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-
scale=1.0">
  <title>Beranda Perpustakaan</title>
  <style>
    * {
      margin: 0;
      padding: 0;
      box-sizing: border-box;
    }

    body {
      font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;
      background-color: #f9f9f9;
    }

    header {
      background-color: #0fa94c;
      padding: 20px;
      text-align: center;
      color: white;
    }

    header h1 {
      font-size: 36px;
      margin-bottom: 10px;
    }

    header p {
      font-size: 18px;
    }

    .container {
      max-width: 1100px;
      margin: 30px auto;
      padding: 20px;
    }
```

```
.content {  
  display: flex;  
  flex-wrap: wrap;  
  justify-content: space-between;  
}
```

```
.card {  
  background-color: rgb(181, 237, 211);  
  box-shadow: 0 4px 8px rgba(0, 0, 0, 0.1);  
  border-radius: 10px;  
  width: 48%;  
  margin-bottom: 20px;  
  padding: 20px;  
  text-align: justify;  
}
```

```
.card h3 {  
  font-size: 24px;  
  color: #343a40;  
  margin-bottom: 15px;  
}
```

```
.card p {  
  text-align: justify;  
  font-size: 16px;  
  color: #555555;  
  line-height: 1.6;  
}
```

```
.button {  
  display: inline-block;  
  padding: 10px 20px;  
  background-color: #28a745;  
  color: white;  
  text-align: center;  
  border-radius: 5px;  
  text-decoration: none;  
  font-size: 16px;  
  font-weight: bold;  
  margin-top: 15px;  
}
```

```

.button:hover {
    background-color: #218838;
}

footer {
    text-align: center;
    margin-top: 30px;
    padding: 20px;
    background-color: #0fa94c;
    color: white;
}

footer p {
    font-size: 14px;
}

.back-button {
    display: inline-block;
    padding: 12px 25px;
    background-color: #5df98c; /* Warna tombol */
    color: rgb(0, 0, 0);
    text-align: center;
    border-radius: 30px; /* Membuat tombol menjadi bulat */
    text-decoration: none;
    font-size: 18px;
    font-weight: bold;
    transition: background-color 0.3s ease, transform 0.2s ease; /*
Animasi */
    box-shadow: 0 8px 15px rgba(0, 0, 0, 0.1); /* Efek bayangan */
}

.back-button:hover {
    background-color: #36a649; /* Warna tombol saat di-hover */
    transform: translateY(-3px); /* Efek bergerak ke atas saat di-hover */
    box-shadow: 0 12px 20px rgba(0, 0, 0, 0.2); /* Bayangan saat hover */
}

```

```

.back-button:active {
    transform: translateY(0px); /* Menormalkan posisi saat diklik */
    box-shadow: 0 6px 10px rgba(0, 0, 0, 0.15); /* Mengurangi
bayangan saat klik */
}
</style>
</head>
<body>
    <header>
        <h1>Selamat Datang di Perpustakaan Digital</h1>
        <p>Menyediakan akses mudah ke ribuan buku dan referensi secara
online</p>
    </header>

    <div class="container">
        <div class="content">
            <!-- Card 1 -->
            <div class="card">
                <h3>Koleksi Buku Terlengkap</h3>
                <p>Kami menyediakan berbagai buku dari beragam kategori,
mulai dari fiksi, non-fiksi, sains, sejarah, teknologi, hingga referensi
akademik.
                Temukan buku favorit Anda dan perluas wawasan dengan
koleksi kami.</p>
            </div>
            <!-- Card 2 -->
            <div class="card">
                <h3>Layanan Keanggotaan</h3>
                <p>Jadilah anggota perpustakaan untuk mendapatkan akses
penuh ke koleksi kami, peminjaman buku, dan berbagai manfaat lainnya.
Keanggotaan terbuka untuk umum, baik pelajar, mahasiswa, maupun
masyarakat umum.</p>
            </div>

```

```

<!-- Card 3 -->
<div class="card">
  <h3>Acara dan Workshop</h3>
  <p>Kami secara rutin mengadakan acara seperti diskusi buku,
seminar, dan workshop.
    Ayo ikuti acara kami untuk menambah wawasan dan berbagi
    pengetahuan dengan komunitas lainnya.</p>
</div>
<!-- Card 4 -->
<div class="card">
  <h3>Bantuan dan Dukungan</h3>
  <li><strong>Email:</strong>
support@perpustakaan.com</li>
  <li><strong>Telepon:</strong> (021) 123-4567</li>
  <li><strong>Datang langsung ke: </strong>
    <br>Jl. Medan Merdeka Sel. No.11 · 0857-1714-
7303
  </li>
</div>
</div>
</div>
<!-- Button Kembali -->
<a href="/auth/login" class="back-button">Kembali ke Halaman
Awal</a>
</div>

<footer>
  <p>&copy; 2024 Perpustakaan Digital. Semua hak dilindungi. |
  Email: support@perpustakaan.com | Telp: (021) 123-4567</p>
</footer>
</body>
</html>

```

7. Sebelum konfigurasi buat kode aplikasi yang dimana file server utama node.js

Source code app.js

```

const express = require('express');
const bodyParser = require('body-parser');
const session = require('express-session');
const authRoutes = require('./routes/auth');
const path = require('path');
const app = express();

```

```

// Set EJS sebagai template engine
app.set('view engine', 'ejs');

// Middleware
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: true }));
app.use(session({
  secret: 'secret',
  resave: false,
  saveUninitialized: true
}));

// Set static folder
app.use(express.static(path.join(__dirname, 'public')));

// Middleware to check login status
app.use((req, res, next) => {
  if (!req.session.user && req.path !== '/auth/login' && req.path !==
'/auth/register') {
    //If the user is not logged in and trying to access any other page except
login/register
    return res.redirect('/auth/login');
  } else if (req.session.user && req.path === '/') {
    //If user is logged in and tries to access the root route, redirect to profile
    return res.redirect('/auth/profile');
  }
  next();
});

// Routes
app.use('/auth', authRoutes);

// Root Route: Redirect to /auth/login or /auth/profile based on session
app.get('/', (req, res) => {
  if (req.session.user) {
    return res.redirect('/auth/profile');
  } else {
    return res.redirect('/auth/login');
  }
});

```

```
// Menjalankan Server
app.listen(3000, () => {
  console.log('Server running on port 3000');
});
```

8. Selanjutnya untuk konfigurasi dengan db.js

Source code db.js

```
const mysql = require('mysql');

const db = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'library'
});

db.connect((err) => {
  if (err) {
    throw err;
  }
  console.log('MySQL connected');
});

module.exports = db;
```

9. Setelah itu konfigurasi rute servernya dengan buat folder routes input file auth.js

Source code auth.js

```
const express = require('express');
const router = express.Router();
const bcrypt = require('bcryptjs');
const db = require('../config/db');

//Render halaman register
router.get('/register', (req, res) => {
  res.render('register');
});
```

```

//Proses register user
router.post('/register', (req, res) => {
  const { username, email, password } = req.body;
  const hashedPassword = bcrypt.hashSync(password, 10);
  const query = "INSERT INTO admins (username, email, password)
VALUES (?, ?, ?)";
  db.query(query, [username, email, hashedPassword], (err, result) => {
    if(err) throw err;
    res.redirect('/auth/login');
  });
});

// Render halaman login
router.get('/login', (req, res) => {
  res.render('login');
});

//Proses login user
router.post('/login', async (req, res) => {
  const { username, password } = req.body;

  const query = "SELECT * FROM admins WHERE username = ?";
  db.query(query, [username], (err, result) => {
    if(err) throw err;
    if (result.length > 0){
      const user = result[0];
      if (bcrypt.compareSync(password, user.password)) {
        req.session.user = user;
        res.redirect('/auth/profile');
      } else {
        res.send('Incorrect password');
      }
    } else {
      res.send('User not found');
    }
  });
});

```

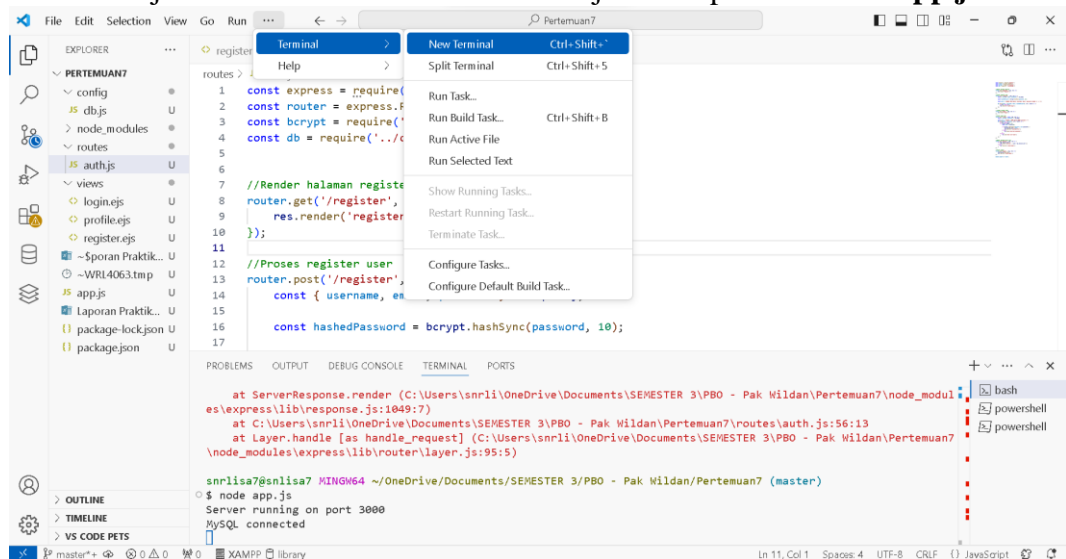


```
//Render halaman profil user
router.get('/profile', (req, res) => {
  if (req.session.user){
    res.render('profile', { user: req.session.user });
  } else {
    res.redirect('/auth/login');
  }
});
```

```
//Proses Logout
router.get('/logout', (req, res) => {
  req.session.destroy();
  res.redirect('/auth/login');
});
```

```
module.exports = router;
```

10. Untuk menjalankan server buka terminal dan jalankan perintah **node app.js**

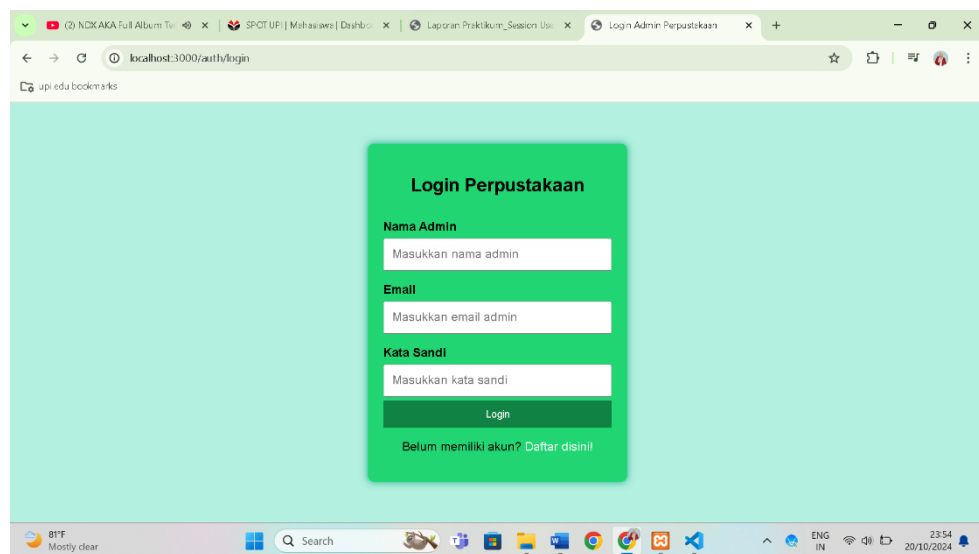


Kemudian server akan berjalan diport 3000 dan dapat mengakses halaman web di browser melalui URL : <http://localhost:3000>.

V. HASIL DAN PEMBAHASAN

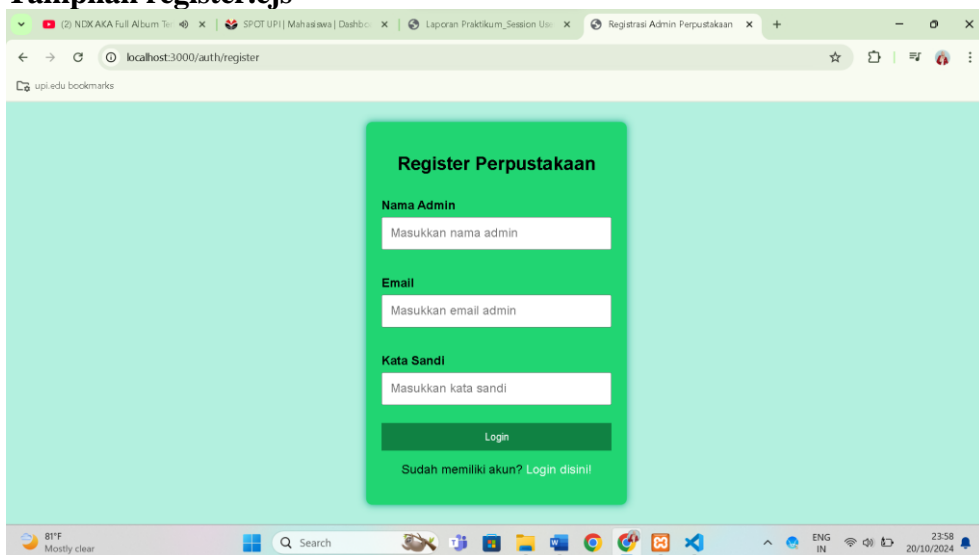
Setelah server dijalankan, halaman login.ejs akan tampil di browser ketika mengakses `http://localhost:3000`. Proses ini menunjukkan bagaimana server Node.js membaca file dari sistem dan mengirimkannya sebagai tanggapan ke browser. Dalam praktik ini, saya berhasil membuat server sederhana dan memahami bagaimana data diambil dari file sistem serta ditampilkan sebagai halaman web.

Tampilan login.ejs



The screenshot shows a web browser window with the address bar displaying `localhost:3000/auth/login`. The page features a light blue background with a central green login form titled "Login Perpustakaan". The form includes three input fields: "Nama Admin" (Masukkan nama admin), "Email" (Masukkan email admin), and "Kata Sandi" (Masukkan kata sandi). Below these fields is a green "Login" button. At the bottom of the form, there is a link that says "Belum memiliki akun? Daftar disini!". The browser's taskbar at the bottom shows the system clock as 23:54 on 20/10/2024.

Tampilan register.ejs

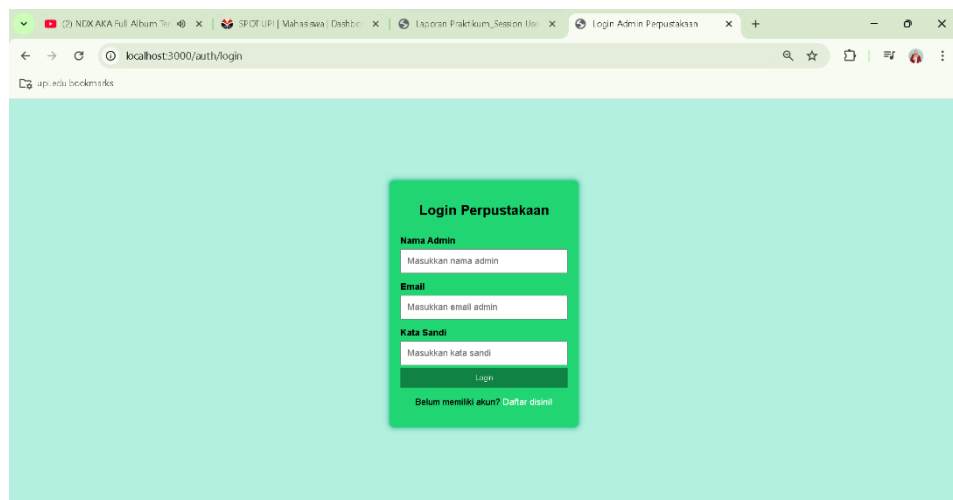


The screenshot shows a web browser window with the address bar displaying `localhost:3000/auth/register`. The page features a light blue background with a central green registration form titled "Register Perpustakaan". The form includes three input fields: "Nama Admin" (Masukkan nama admin), "Email" (Masukkan email admin), and "Kata Sandi" (Masukkan kata sandi). Below these fields is a green "Login" button. At the bottom of the form, there is a link that says "Sudah memiliki akun? Login disini!". The browser's taskbar at the bottom shows the system clock as 23:58 on 20/10/2024.

Tampilan profile.ejs ketika setelah berhasil login



Ketika diklik menu kembali ke halaman awal akan kembali ke menu login



VI. KESIMPULAN

Praktikum ini telah berhasil menghasilkan aplikasi manajemen perpustakaan sederhana yang berfungsi untuk registrasi dan autentikasi admin. Penggunaan teknologi seperti Node.js, Express, bcryptjs, body-parser, express-session, dan EJS memudahkan dalam membangun aplikasi web yang dinamis dan aman. Aplikasi ini dapat dikembangkan lebih lanjut dengan menambahkan fitur-fitur lain seperti pengelolaan buku, peminjaman, dan pengembalian.