

Laporan Praktikum
Mata Kuliah Pemrograman WEB



Pertemuan 6. Tugas 1

Dosen Pengampu :

Willdan Aprizal Arifin, S.Pd., M.Kom.

Disusun Oleh :

(Novia Ramadhani)

(2305968)

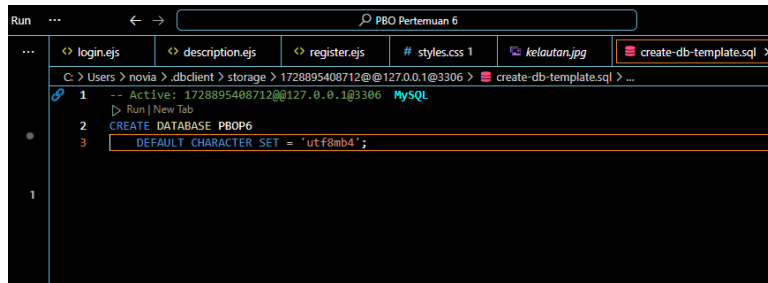
PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA

2024

SESSION

Session adalah mekanisme yang digunakan dalam aplikasi web untuk menyimpan informasi pengguna secara sementara saat mereka berinteraksi dengan aplikasi tersebut. Dalam konteks Node js, session digunakan untuk melacak dan mengingat status pengguna di antara permintaan HTTP (request) yang berbeda. Disini saya menambahkan Table Description dan mengubah sedikit tampilannya

1. Membuat database dengan nama PBO6



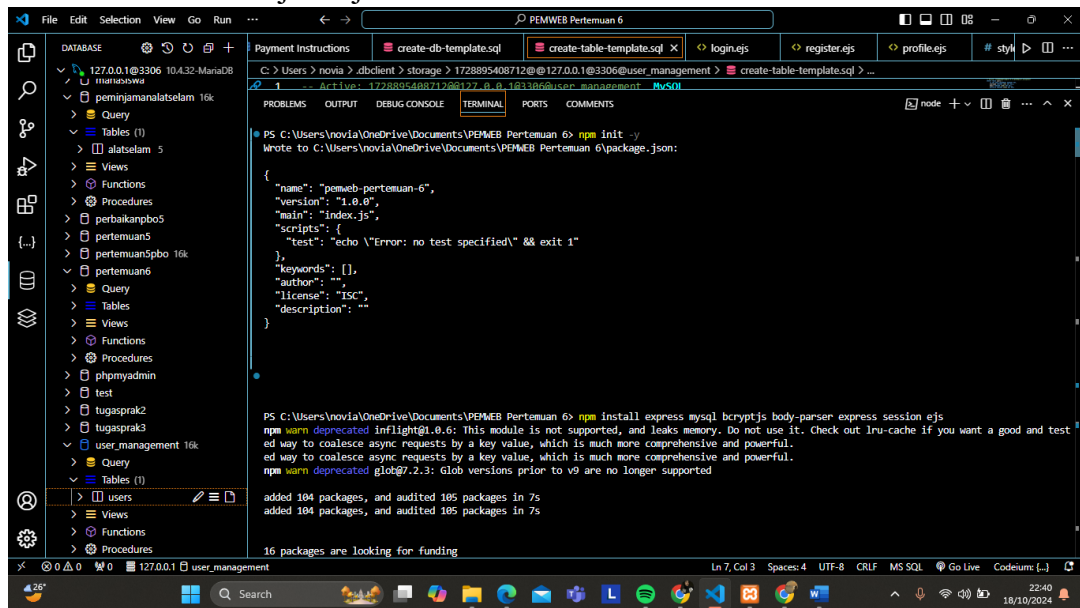
```
Run ... PBO Pertemuan 6
... login.ejs description.ejs register.ejs # styles.css 1 kelautan.jpg create-db-template.sql X
C: > Users > novia > .dbclient > storage > 1728895408712@@127.0.0.1@3306 > create-db-template.sql > ...
1 -- Active: 1728895408712@@127.0.0.1@3306 MySQL
  > Run | New Tab
2 CREATE DATABASE PBO6
3 DEFAULT CHARACTER SET = 'utf8mb4';
```

2. Membuat table users dengan struktur berikut



```
View Go Run ... PEMWEB Pertemuan 6
Payment Instructions create-db-template.sql create-table-template.sql X login.ejs register
3306 10.4.32-MariaDB
analatselam 16k
1 -- Active: 1728895408712@@127.0.0.1@3306@user_management MySQL
  > Run | Reset
2 CREATE TABLE users (
3   id INT AUTO_INCREMENT PRIMARY KEY,
4   username VARCHAR (50) NOT NULL,
5   email VARCHAR (100) NOT NULL,
6   password VARCHAR (255) NOT NULL
7 ); 17ms
```

3. Inisialisasi Node.js Project

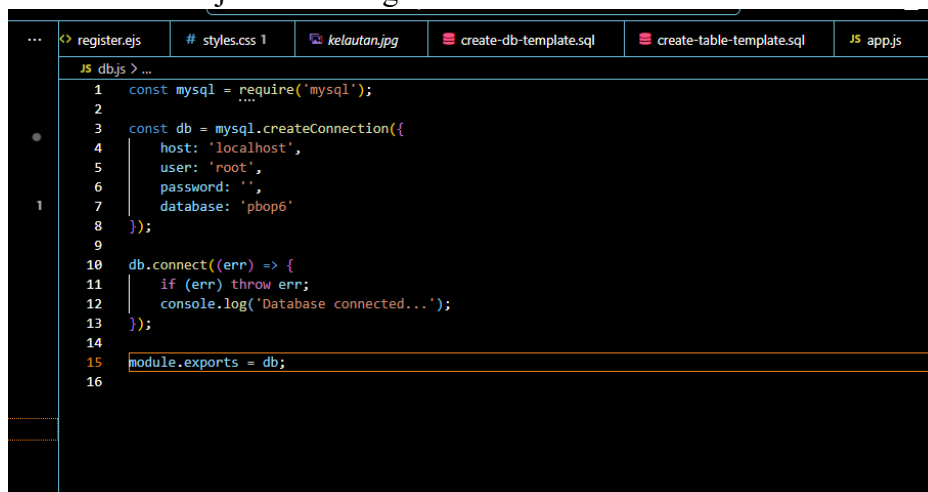


```
PS C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6> npm init -y
Wrote to C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6\package.json:

{
  "name": "pemweb-pertemuan-6",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "license": "ISC",
  "description": ""
}

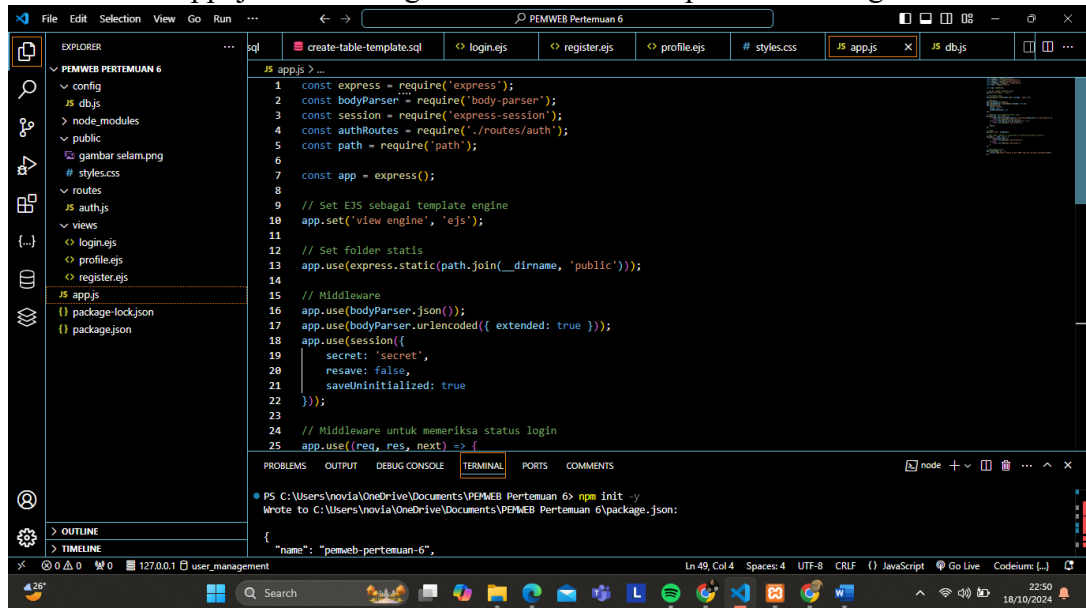
PS C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6> npm install express mysql bcryptjs body-parser express-session
npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you want a good and test
ed way to coalesce async requests by a key value, which is much more comprehensive and powerful.
npm warn deprecated glob@7.2.3: Glob versions prior to v9 are no longer supported
added 104 packages, and audited 105 packages in 7s
added 104 packages, and audited 105 packages in 7s
16 packages are looking for funding
```

4. Buat file db.js untuk mengatur koneksi ke database



```
1 const mysql = require('mysql');
2
3 const db = mysql.createConnection({
4   host: 'localhost',
5   user: 'root',
6   password: '',
7   database: 'pbop6'
8 });
9
10 db.connect((err) => {
11   if (err) throw err;
12   console.log('Database connected...');
13 });
14
15 module.exports = db;
```

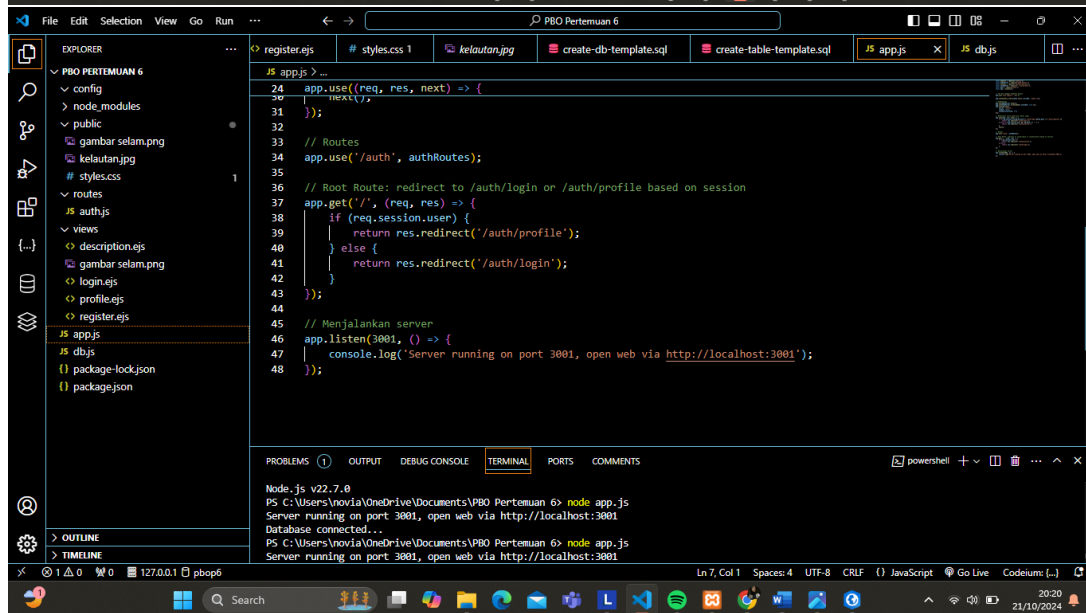
5. Buat file app.js untuk menginisialisasi server Express dan mengatur middleware



```
File Edit Selection View Go Run ...
PEMWEB Pertemuan 6
create-table-template.sql login.ejs register.ejs profile.ejs styles.css app.js db.js

EXPLOLERER
PEMWEB PERTEMUAN 6
  config
  db.js
  node_modules
  public
  gambar_salam.png
  styles.css
  routes
  auth.js
  views
  login.ejs
  profile.ejs
  register.ejs
  app.js
  package-lock.json
  package.json

app.js
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 // Set folder statis
13 app.use(express.static(path.join(__dirname, 'public')));
14
15 // Middleware
16 app.use(bodyParser.json());
17 app.use(bodyParser.urlencoded({ extended: true }));
18 app.use(session({
19   secret: 'secret',
20   resave: false,
21   saveUninitialized: true
22 }));
23
24 // Middleware untuk memeriksa status login
25 app.use((req, res, next) => {
26
27   PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
28
29   PS C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6> npm init -y
30   Wrote to C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6\package.json:
31
32   {
33     "name": "pemweb-pertemuan-6",
34
35   }
36
37 user_management
38 127.0.0.1
39 22:50
40 18/10/2024
```



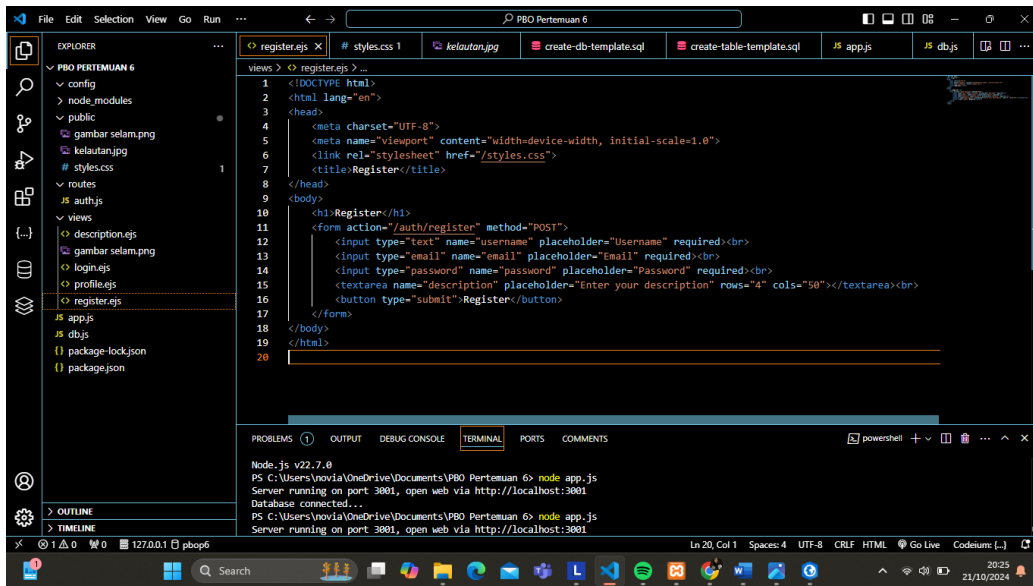
```
File Edit Selection View Go Run ...
PBO Pertemuan 6
register.ejs styles.css 1 kelautan.jpg create-db-template.sql create-table-template.sql app.js db.js

EXPLOLERER
PBO PERTEMUAN 6
  config
  node_modules
  public
  gambar_salam.png
  kelautan.jpg
  styles.css
  routes
  auth.js
  views
  description.ejs
  gambar_salam.png
  login.ejs
  profile.ejs
  register.ejs
  app.js
  db.js
  package-lock.json
  package.json

app.js
24 app.use((req, res, next) => {
25   next();
26 });
27
28 // Routes
29 app.use('/auth', authRoutes);
30
31 // Root Route: redirect to /auth/login or /auth/profile based on session
32 app.get('/', (req, res) => {
33   if (req.session.user) {
34     return res.redirect('/auth/profile');
35   } else {
36     return res.redirect('/auth/login');
37   }
38 });
39
40 // Menjalankan server
41 app.listen(3001, () => {
42   console.log('Server running on port 3001, open web via http://localhost:3001');
43 });
44
45 PROBLEMS 1 OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS
46
47 Node.js v22.7.0
48 PS C:\Users\novia\OneDrive\Documents\PBO Pertemuan 6> node app.js
49 Server running on port 3001, open web via http://localhost:3001
50 Database connected...
51 PS C:\Users\novia\OneDrive\Documents\PBO Pertemuan 6> node app.js
52 Server running on port 3001, open web via http://localhost:3001
53
54 127.0.0.1 pbop6
55 20:20
56 21/10/2024
```

6. Membuat tampilan views

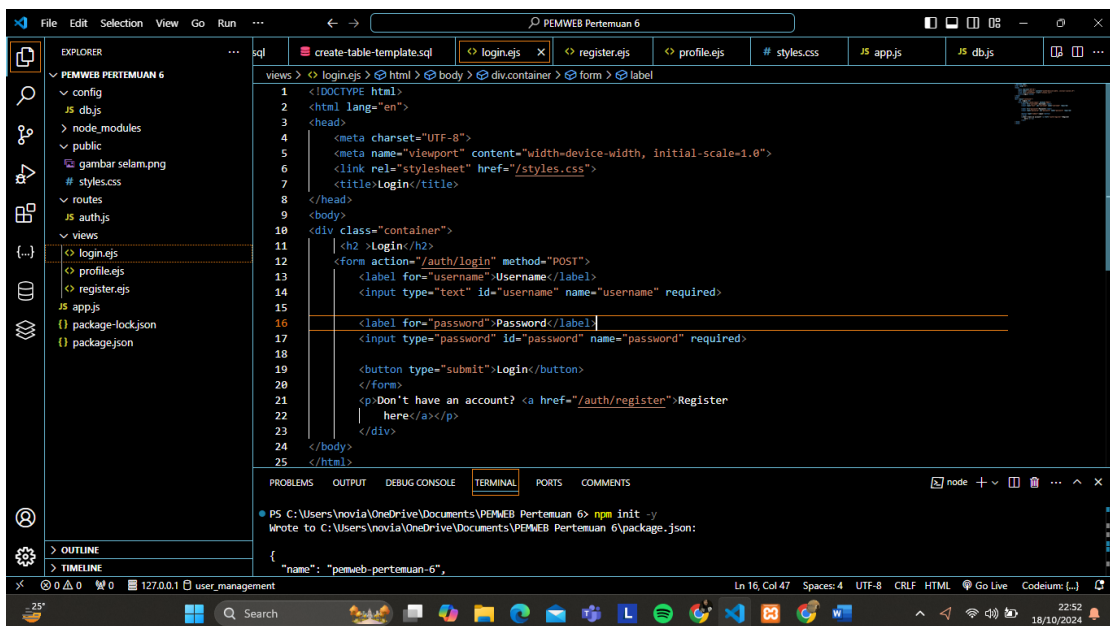
- Register



```
views > register.ejs > ...
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   <h1>Register</h1>
11   <form action="/auth/register" method="POST">
12     <input type="text" name="username" placeholder="Username" required><br>
13     <input type="email" name="email" placeholder="Email" required><br>
14     <input type="password" name="password" placeholder="Password" required><br>
15     <textarea name="description" placeholder="Enter your description" rows="4" cols="50"></textarea><br>
16     <button type="submit">Register</button>
17   </form>
18 </body>
19 </html>
20
```

Node.js v22.7.0
PS C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6> node app.js
Server running on port 3801, open web via http://localhost:3801
Database connected...
PS C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6> node app.js
Server running on port 3801, open web via http://localhost:3801

- Login



```
views > login.ejs > html > body > div.container > form > label
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
22       here</a></p>
23   </div>
24 </body>
25 </html>
```

PS C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6> npm init -y
Wrote to C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6\package.json:
{
 "name": "pemweb-pertemuan-6",
}

- Profile

```
views > < profile.ejs > html > < body
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>
```

```
PS C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6> npm init -y
Wrote to C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6\package.json:
{
  "name": "pemweb-pertemuan-6",
```

- Description

```
views > < description.ejs > html > < body
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>User Description</title>
8 </head>
9 <body>
10   <h1>Profile of <%= user.username %></h1>
11   <p>Email: <%= user.email %></p>
12   <h3>Description</h3>
13   <p><%= user.description ? user.description : 'No description available' %></p>
14   <form action="/auth/update-description" method="POST">
15     <textarea name="description" placeholder="Enter your description here" rows="4" cols="50"><%= user.descr
16   </form>
17   <button type="submit">Update Description</button>
18   <br>
19   <a href="/auth/logout">Logout</a>
20 </body>
21 </html>
```

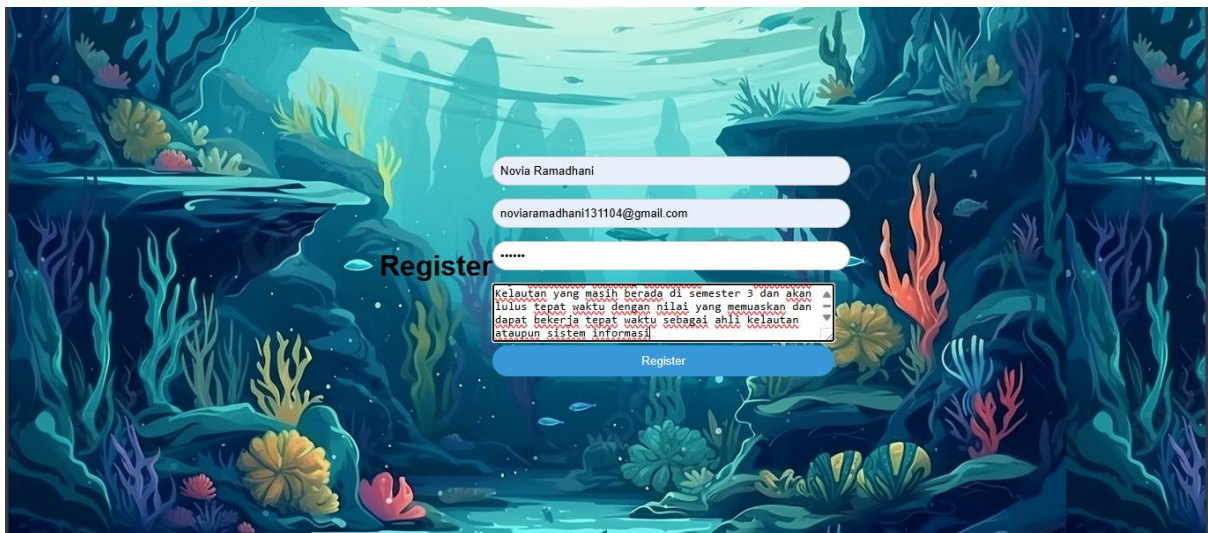
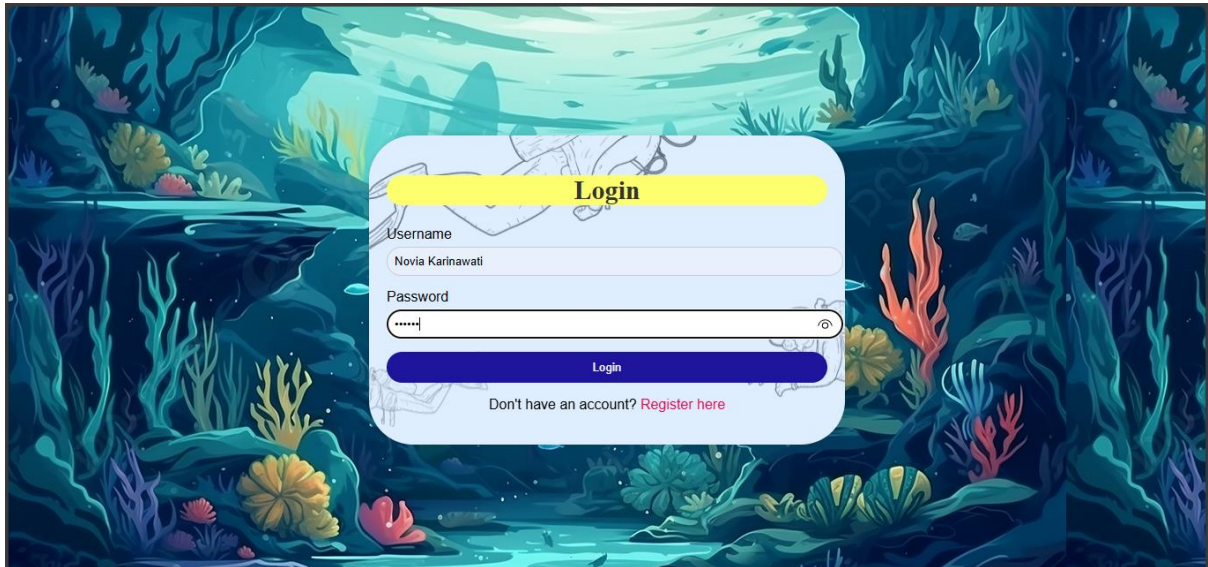
```
Node.js v22.7.0
PS C:\Users\novia\OneDrive\Documents\PBO Pertemuan 6> node app.js
Server running on port 3001, open web via http://localhost:3001
Database connected...
PS C:\Users\novia\OneDrive\Documents\PBO Pertemuan 6> node app.js
Server running on port 3001, open web via http://localhost:3001
```

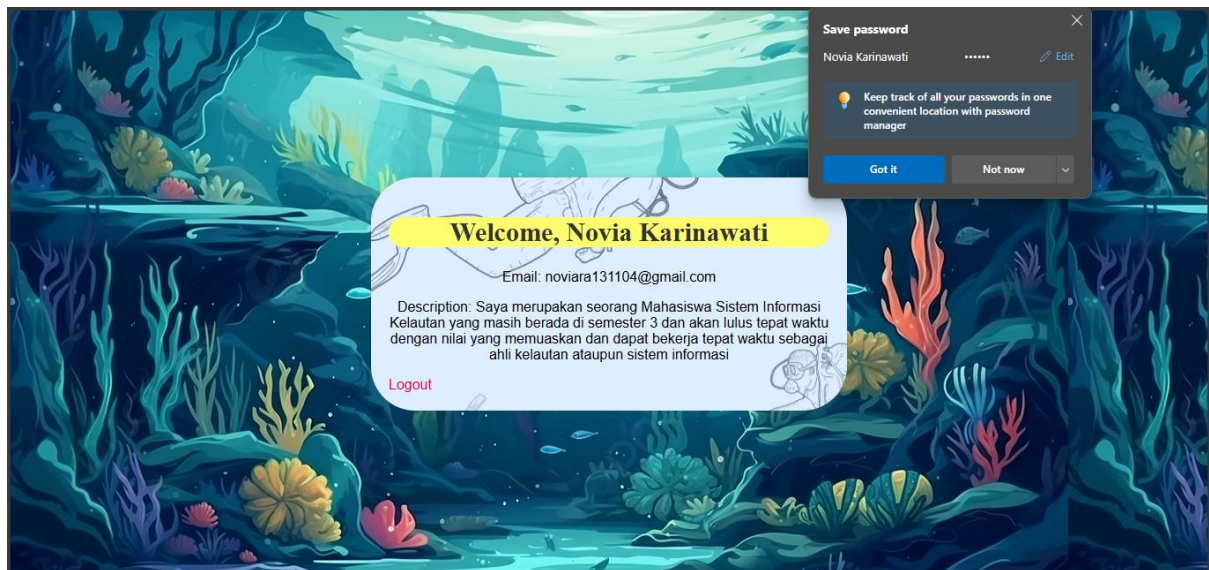
7. Buat file auth.js di dalam folder routes untuk mengelola autentikasi pengguna



```
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   const hashedPassword = bcrypt.hashSync(password, 10);
15
16   const query = "INSERT INTO users (username, email, password) VALUES (?, ?, ?)";
17   db.query(query, [username, email, hashedPassword], (err, result) => {
18     if (err) throw err;
19     res.redirect('/auth/login');
20   });
21 });
22
23 // Render halaman login
24 router.get('/login', (req, res) => {
25   res.render('login');
26 });
27
28 // Proses login user
29 router.post('/login', (req, res) => {
30   const { username, password } = req.body;
31
32   const query = "SELECT * FROM users WHERE username = ?";
33   db.query(query, [username], (err, result) => {
34     if (err) throw err;
35     if (result.length > 0) {
36       const user = result[0];
37
38       if (bcrypt.compareSync(password, user.password)) {
39         req.session.user = user;
40         res.redirect('/auth/profile');
41       } else {
42         res.send('Incorrect password');
43       }
44     } else {
45       res.send('User not found');
46     }
47   });
48 });
49
50 // Proses update description
51 router.post('/update-description', (req, res) => {
52   const { description } = req.body;
53   const userId = req.session.user.id;
54
55   const query = "UPDATE users SET description = ? WHERE id = ?";
56   db.query(query, [description, userId], (err, result) => {
57     if (err) throw err;
58     res.redirect('/auth/profile');
59   });
60 });
61
62 // Proses logout
63 router.get('/logout', (req, res) => {
64   req.session.destroy();
65   res.redirect('/auth/login');
66 });
67
68 module.exports = router;
```

HASIL





Kesimpulan

Praktikum ini berhasil menunjukkan langkah-langkah dalam membangun aplikasi web sederhana menggunakan Node.js dan Express. Proses dimulai dengan inisialisasi proyek Node.js dan pengaturan koneksi ke database melalui file db.js. Selanjutnya, server Express diinisialisasi dalam file app.js, di mana middleware juga diatur untuk menangani permintaan dari pengguna. Aplikasi ini mencakup tampilan untuk registrasi, login, dan profil pengguna, yang dirancang untuk memberikan pengalaman pengguna yang baik. Selain itu, penggunaan CSS untuk mempercantik tampilan aplikasi juga dibahas, dengan hasil yang ditampilkan dalam laporan.

Secara keseluruhan, praktikum ini memberikan pemahaman yang lebih baik tentang pengembangan aplikasi web, pengelolaan autentikasi, dan pentingnya desain antarmuka yang menarik. Dengan demikian, mahasiswa diharapkan dapat menerapkan pengetahuan ini dalam proyek-proyek mendatang dan mengembangkan keterampilan pemrograman web mereka lebih lanjut.