

Laporan Praktikum
UTS PEMWEB dan PBO



Dosen Pengampu:

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

Disusun Oleh:

(Novia Ramadhani)

(2305968)

PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA
2024

Link YOUTUBE: <https://youtu.be/dD8Owd9MKK0>

CRUD

CRUD adalah singkatan dari *Create*, *Read*, *Update*, dan *Delete*, yaitu operasi dasar dalam pengolahan data di database khususnya pada database yang relasional. CRUD dapat diterapkan secara luas di berbagai jenis database seperti MySQL, PostgreSQL, MongoDB, dan sejenisnya.

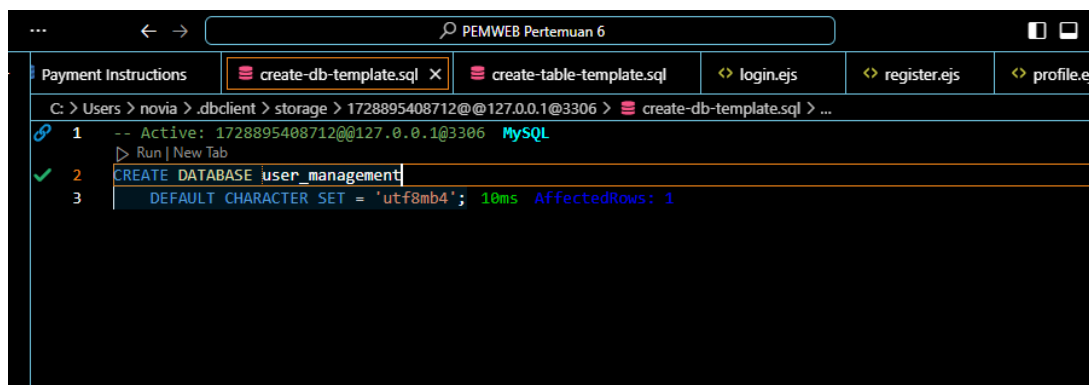
Selain itu, CRUD adalah fondasi dasar dalam pengembangan sistem atau aplikasi yang berhubungan dengan pengolahan data. Ini karena hampir semua aplikasi yang menggunakan database melibatkan operasi CRUD dalam penggunaannya.

SESSION

Session adalah periode waktu yang dimulai ketika seorang pengguna mulai berinteraksi dengan suatu sistem atau aplikasi, dan berakhir ketika pengguna keluar dari sistem atau aplikasi tersebut. Selama sesi, sistem akan menetapkan sebuah “session ID” yang unik untuk mengidentifikasi pengguna yang sedang menggunakan sistem atau aplikasi.

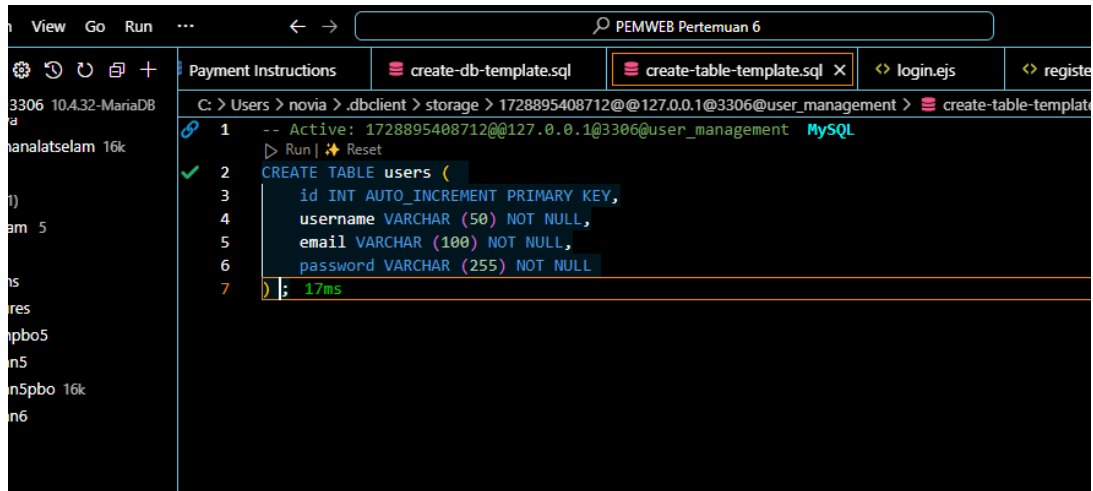
Data session ini disimpan dalam variabel global yang dapat diakses di seluruh aplikasi web, dan dapat digunakan untuk mengidentifikasi pengguna, menyimpan preferensi, dan mempertahankan status login.

Membuat database dengan nama user_management



```
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 user_management  
3 DEFAULT CHARACTER SET = 'utf8mb4'; 10ms AffectedRows: 1
```

1. Membuat table users dengan struktur berikut



The screenshot shows a MySQL command line interface. The prompt is 'C: > Users > novia > .dbclient > storage > 1728895408712@@127.0.0.1@3306@user_management >'. The user has entered the command 'CREATE TABLE users (' followed by the table structure: 'id INT AUTO_INCREMENT PRIMARY KEY,', 'username VARCHAR (50) NOT NULL,', 'email VARCHAR (100) NOT NULL,', and 'password VARCHAR (255) NOT NULL'. The command is completed with a semicolon and a newline. The output shows the command was executed successfully in 17ms.

```
1 -- Active: 1728895408712@@127.0.0.1@3306@user_management MySQL
2 > Run | Reset
3 CREATE TABLE users (
4   id INT AUTO_INCREMENT PRIMARY KEY,
5   username VARCHAR (50) NOT NULL,
6   email VARCHAR (100) NOT NULL,
7   password VARCHAR (255) NOT NULL
8 );
9 17ms
```

2. Inisialisasi Node.js Project



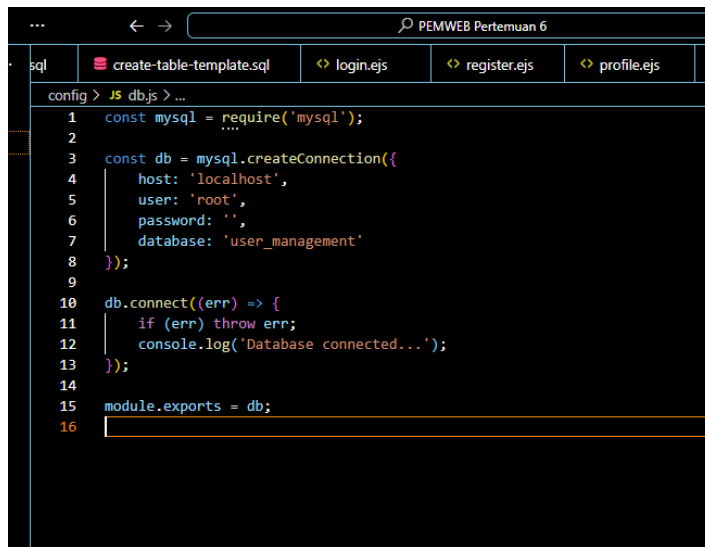
The screenshot shows a Node.js terminal window. The prompt is 'PS C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6>'. The user has entered the command 'npm init -y'. The output shows the command was executed successfully, creating a package.json file. The package.json file contains the following information: '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 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": ""
}

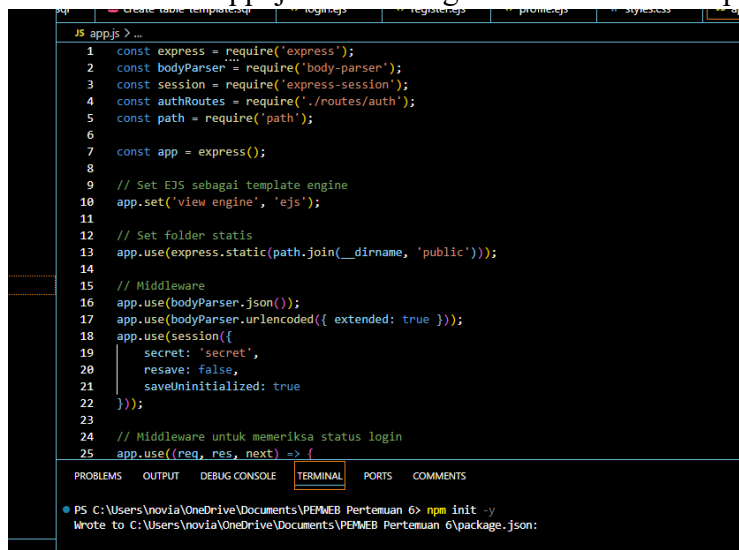
PS C:\Users\novia\OneDrive\Documents\PEMWEB Pertemuan 6> npm install express mysql bcryptjs body-parser express-session ejs
npm warn deprecated inflight@1.0.6: This module is not supported, and leaks memory. Do not use it. Check out lru-cache if you
ed way to coalesce async requests by a key value, which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

3. 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: 'user_management'
8 });
9
10 db.connect((err) => {
11   if (err) throw err;
12   console.log('Database connected...');
13 });
14
15 module.exports = db;
```

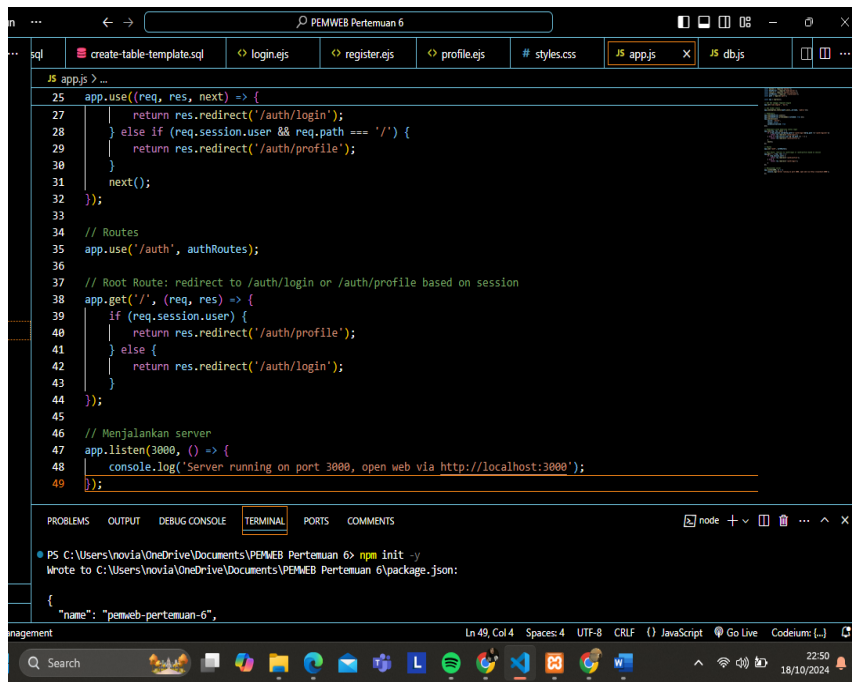
4. Buat file app.js untuk menginisialisasi server Express dan mengatur middleware



```
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) => {
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS

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



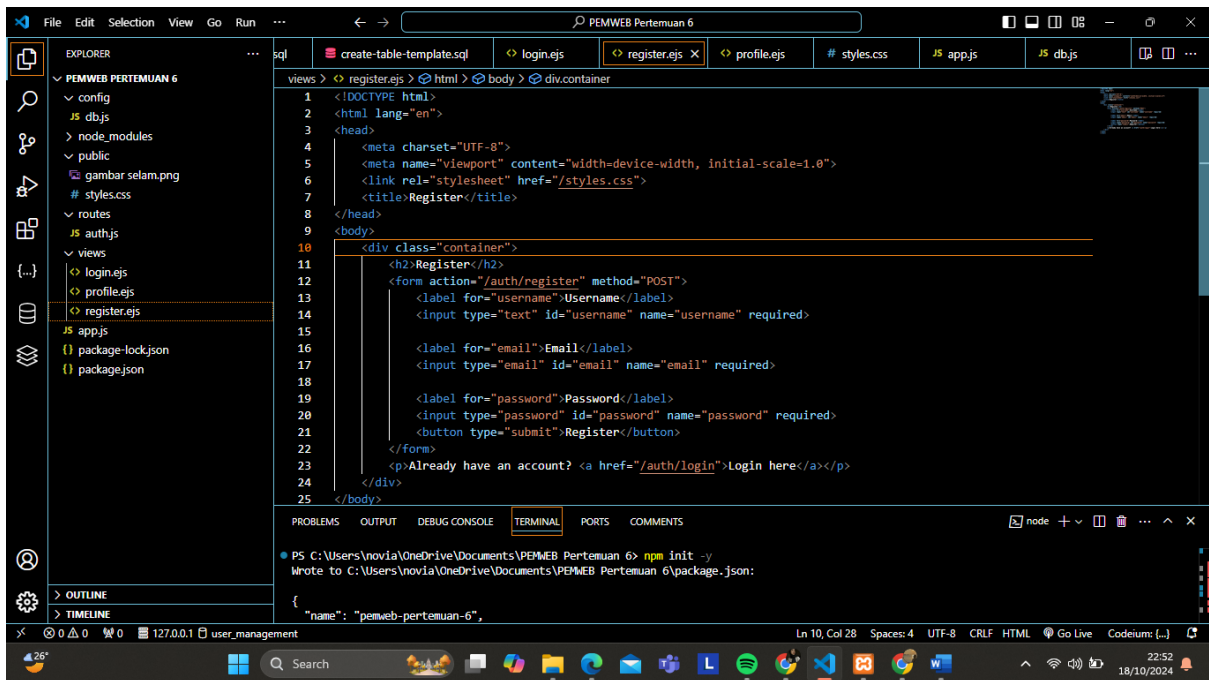
```
app.js ...
25 app.use((req, res, next) => {
26   return res.redirect('/auth/login');
27 } else if (req.session.user && req.path === '/') {
28   return res.redirect('/auth/profile');
29 }
30 next();
31 });
32
33 // Routes
34 app.use('/auth', authRoutes);
35
36 // Root Route: redirect to /auth/login 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 });
44
45 // Menjalankan server
46 app.listen(3000, () => {
47   console.log('Server running on port 3000, open web via http://localhost:3000');
48 });
```

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",
```

5. Membuat tampilan views

- Register



```
views > register.ejs > html > body > div.container
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       <button type="submit">Register</button>
22     </form>
23     <p>Already have an account? <a href="/auth/login">Login here</a></p>
24   </div>
25 </body>
```

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",
```

- 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
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",
```

6. 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');
```

```
... PEMWEB Pertemuan 6
login.js register.js profile.js # styles.css JS app.js JS db.js

routes > JS auth.js > ...
24 router.get('/login', (req, res) => {
25   // Proses login user
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 });

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS COMMENTS JS app.js JS db.js

routes > JS auth.js > ...
29 router.post('/login', (req, res) => {
33   db.query(query, [username], (err, result) => {
45     }
46     res.send('User not found');
47   });
48 });
49
50 // Render halaman profil user
51 router.get('/profile', (req, res) => {
52   if (req.session.user) {
53     res.render('profile', { user: req.session.user });
54   } else {
55     res.redirect('/auth/login');
56   }
57 });
58
59 // Proses logout
60 router.get('/logout', (req, res) => {
61   req.session.destroy();
62   res.redirect('/auth/login');
63 });
64
65 module.exports = router;
66
```

7. Membuat routing CRUD

```
create-table-template.sql edit.js # style.css {} package-lock.json JS app.js X perbaikan

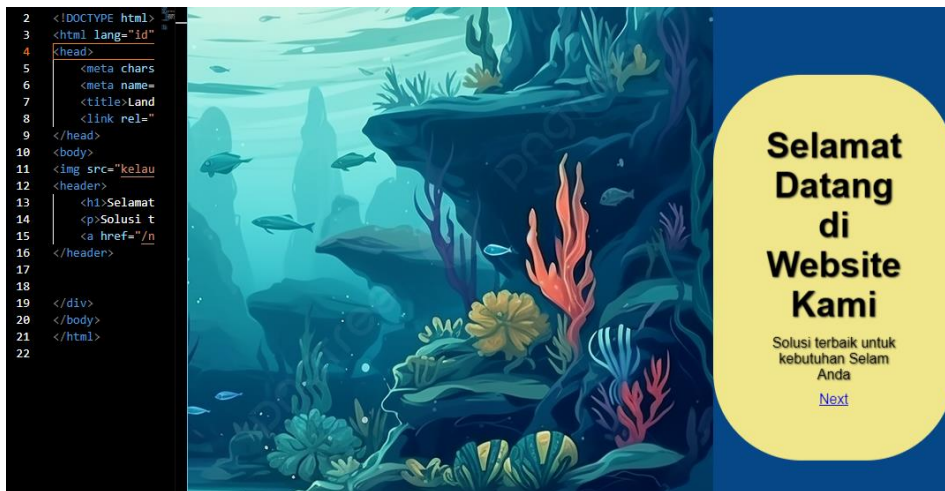
JS app.js > app.post('/update/:id') callback
29 //read
30 app.get('/', (req, res) => {
31   const query = 'SELECT * FROM users';
32   connection.query(query, (err, results) => {
33     res.render('index', { users: results });
34   });
35 });
36
37 //create /input /insert
38 app.post('/add', (req, res) => {
39   const { name, email, phone } = req.body;
40   const query = 'INSERT INTO users (name, email, phone) VALUES (?, ?, ?)';
41   connection.query(query, [name, email, phone], (err, result) => {
42     if (err) throw err;
43     res.redirect('/');
44   });
45 });
46
47 //update
48 //untuk akses halaman
49 app.get('/edit/:id', (req, res) => {
50   const query = 'SELECT * FROM users WHERE id = ?';
51   connection.query(query, [req.params.id], (err, result) => {
52     if (err) throw err;
53     res.render('edit', { user: result[0] });
54   });
55 });
56
57 //untuk update data
58 app.post('/update/:id', (req, res) => {
59   const { name, email, phone } = req.body;
60   const query = 'UPDATE users SET name = ?, email = ?, phone = ? WHERE id = ?';

```

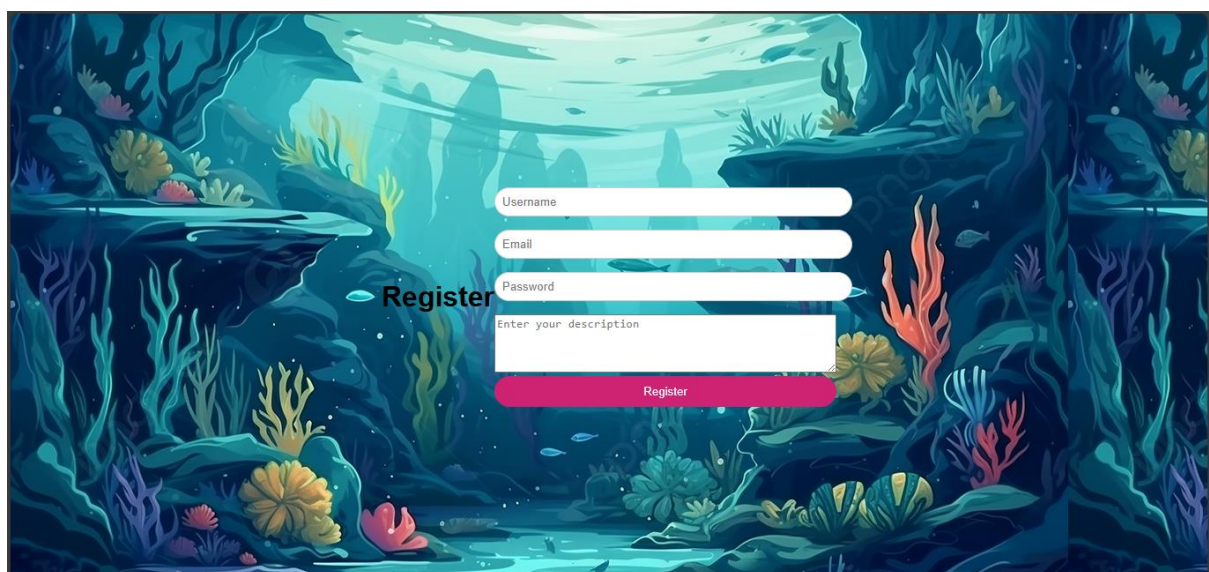
8. Membuat CSS

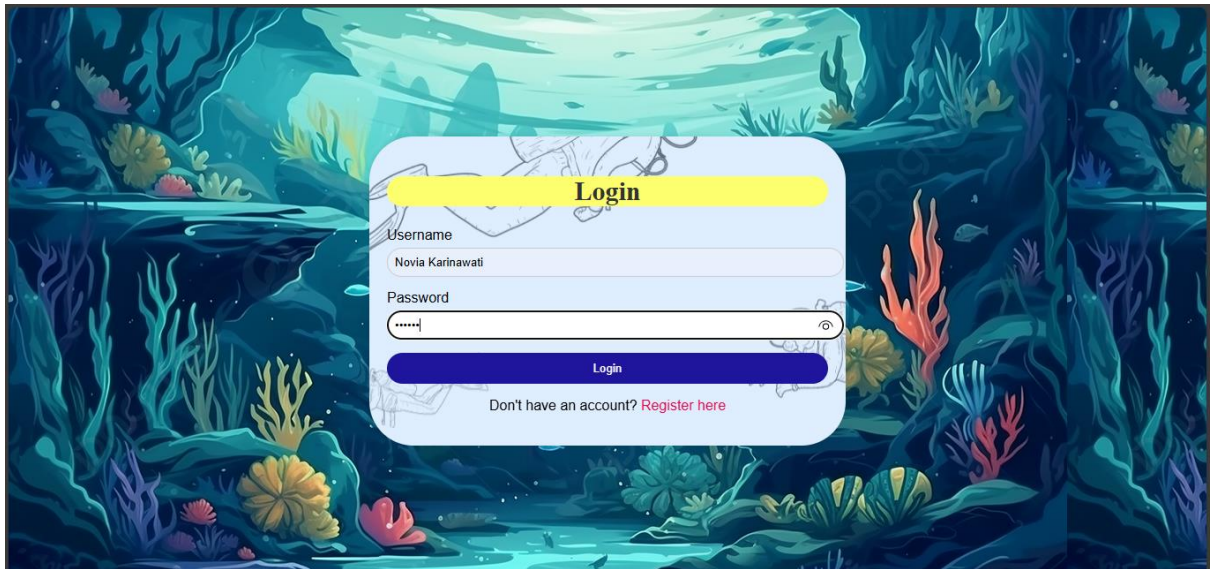
```
public > # styles.css > .container
1 body {
2   font-family: Arial, sans-serif;
3   background-image: url(gambar\selam.png);
4   background-color: #ffcf17;
5   display: flex;
6   justify-content: center;
7   align-items: center;
8   height: 100vh;
9   margin: 0;
10 }
11
12 .container {
13   background-color: #bedcff;
14   background-image: url(gambar\selam.png);
15   padding: 20px;
16   border-radius: 50px;
17   box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);
18   width: 500px;
19 }
20
21 h2 {
22   text-align: center;
23   font-family: 'Times New Roman', Times, serif;
24   background-color: rgb(255, 190, 233);
25   border-radius: 50px;
26 }
27
28 label {
29   display: block;
30   margin-bottom: 5px;
31 }
32
33 input {
34   width: 100%;
35   padding: 8px;
36   margin-bottom: 15px;
37   border: 1px solid #ccc;
38   border-radius: 50px;
39 }
40
41 button {
42   width: 100%;
43   font-size: 18px;
44   padding: 10px;
45 }
46
47 button {
48   font-size: 18px;
49   padding: 10px;
50   background-color: #3998d8;
51   color: white;
52   border: none;
53   border-radius: 50px;
54   cursor: pointer;
55 }
56
57 button:hover {
58   background-color: #1e159c;
59 }
60
61 p {
62   text-align: center;
63 }
64
65 a {
66   color: #e90046;
67   text-decoration: none;
68 }
69
70 a:hover {
71   text-decoration: underline;
72 }
```

9. Membuat Landing Page



Hasil dari CSSnya





Daftar Alat Selam

ID	Nama Alat	Kategori	Kondisi	Stok	Aksi
2	Fins	M	rusak	7	Edit Hapus
4	Wetsuit	M	baik	5	Edit Hapus
5	Mask	S	baik	10	Edit Hapus
8	Snorkel	M	baik	10	Edit Hapus

Tambah Alat Baru

Nama Alat:

Kategori:

Kondisi:

Stok:

Edit Data Alat Selam

Nama Alat:

Kategori:

Kondisi:

Stok:

[Update](#)

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.