

Laporan Praktikum
Mata Kuliah Pemrograman Web



Pertemuan 6
“Sesion”

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

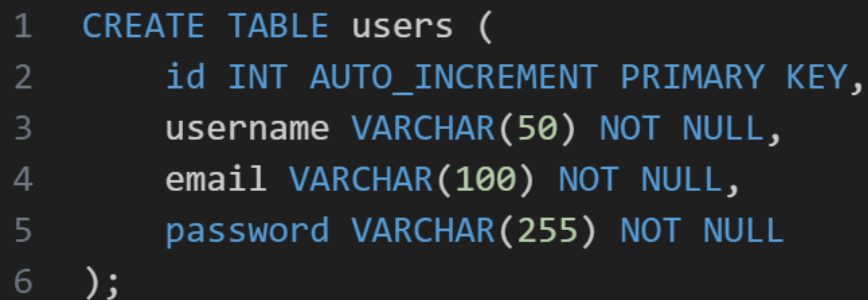
Disusun Oleh :
Fadli Kurnia Ramadhan (2307373)

PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA
2024

I. PENDAHULUAN

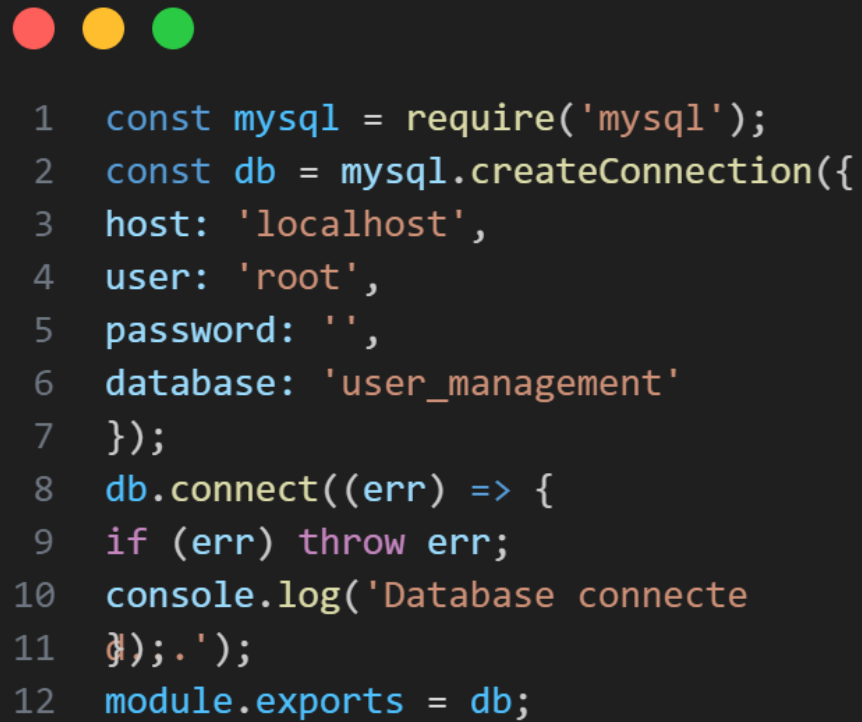
Pada aplikasi berbasis web, komunikasi antara klien (pengguna) dan server harus dikelola dengan hati-hati untuk memastikan sesi tetap aman dan terjaga. Salah satu metode yang sering digunakan adalah manajemen sesi menggunakan session ID. Ketika pengguna masuk (login) ke dalam sistem, server membuat sesi yang unik, dan ID sesi tersebut dikirim ke browser pengguna. Pada setiap permintaan berikutnya, session ID dikirim kembali ke server untuk mengidentifikasi pengguna. Proses ini memastikan bahwa hanya pengguna yang berhak dapat mengakses konten tertentu tanpa harus melakukan autentikasi ulang pada setiap permintaan.

II. PENJELASAN PROGRAM



```
1 CREATE TABLE users (  
2     id INT AUTO_INCREMENT PRIMARY KEY,  
3     username VARCHAR(50) NOT NULL,  
4     email VARCHAR(100) NOT NULL,  
5     password VARCHAR(255) NOT NULL  
6 );
```

1. Langkah awal yang saya buat yaitu membuat data base users. Yang saya pahami pada praktikum kali ini yaitu menambahkan bcryptjs yang berfungsi untuk hashing password, data tidak boleh langsung muncul aslinya agar aman.



```
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 connecte
11   d');
12   module.exports = db;
```

2. Selanjutnya membuat file db.js yang akan menjadi database kita

```

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   } else if (req.session.user && req.path === '/') {
30     // If user is logged in and tries to access the root route, redirect to profile
31     return res.redirect('/auth/profile');
32   }
33   next();
34 });
35
36 // Routes
37 app.use('/auth', authRoutes);
38
39 // Root Route: Redirect to /auth/login or /auth/profile based on session
40 app.get('/', (req, res) => {
41   if (req.session.user) {
42     return res.redirect('/auth/profile');
43   } else {
44     return res.redirect('/auth/login');
45   }
46 });
47
48 // Menjalankan Server
49 app.listen(3000, () => {
50   console.log('Server running on port 3000 http://localhost:3000');
51 });
52

```

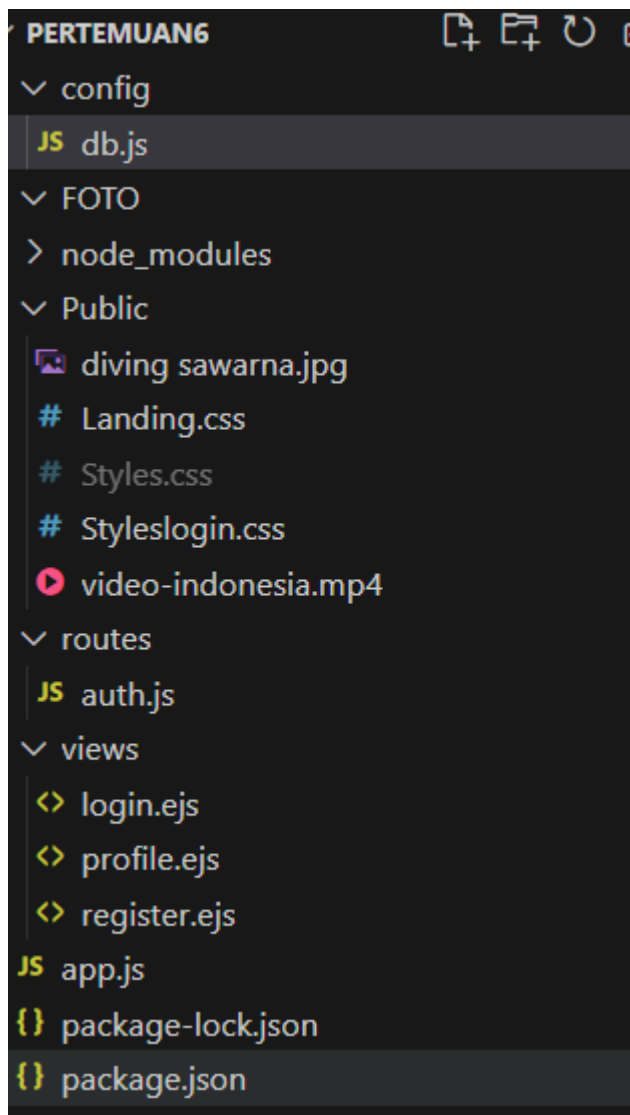
3. Program diatas merupakan server yang akan kita gunakan untuk memanggi informasi-informasi dilain folder.

```

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   const query = "INSERT INTO users (username, email, password) VALUES (?, ?, ?)";
16
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', { video: '<video autoplay muted loop id="background-video"><source src="video-indonesia.mp4" type="video/mp4"></video>'
26 });
27
28 // Proses login user
29 router.post('/login', (req, res) => {
30   const { username, password } = req.body;
31   const query = "SELECT * FROM users WHERE username = ?";
32
33   // Query ke database untuk menemukan user berdasarkan username
34   db.query(query, [username], (err, result) => {
35     if (err) throw err;
36
37     if (result.length > 0) {
38       const user = result[0];
39
40       // Cek apakah password sesuai dengan hash yang ada di database
41       if (bcrypt.compareSync(password, user.password)) {
42         req.session.user = user; // Set sesi user
43         res.redirect('/auth/profile'); // Redirect ke profil jika login berhasil
44       } else {
45         res.send('Incorrect password');
46       }
47     } else {
48       res.send('User not found');
49     }
50   });
51 });
52
53 // Render halaman profil user
54 router.get('/profile', (req, res) => {
55   if (req.session.user) {
56     res.render('profile', { user: req.session.user });
57   } else {
58     res.redirect('/auth/login');
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;
69
70

```

4. Autentifikasi berdasarkan hasil register dan cara kita untuk menginput data di tampilan ejs nantinya.



5. Tampilan diatas merupakan file dan folder yang sudah saya buat.



Login

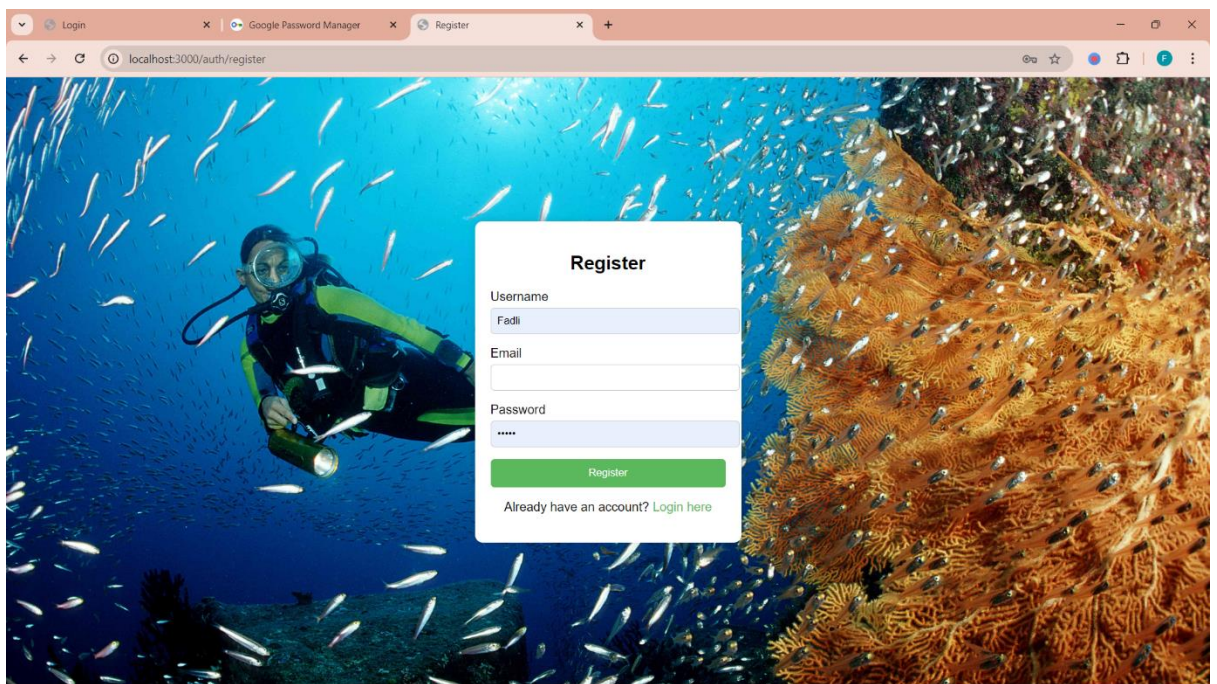
Username
Fadli

Password

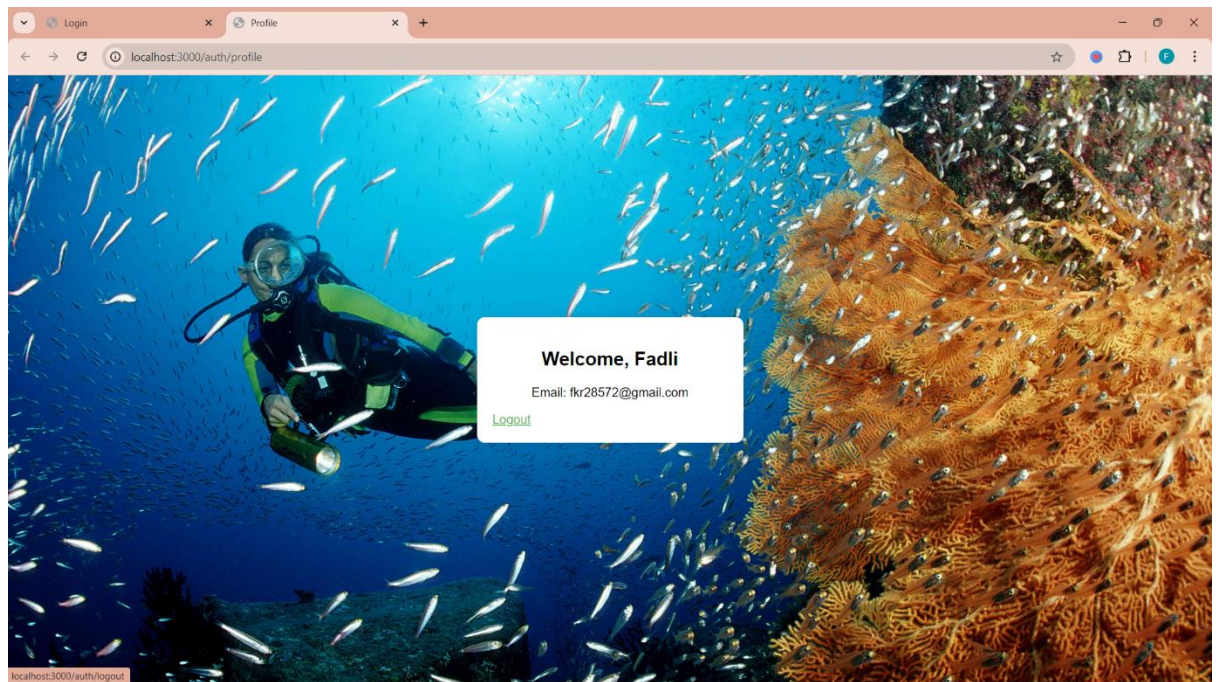
Login

Don't have an account? [Register here](#)

6. Hasil dari menu “Login”.



7. Hasil dari menu “Register”.



8. Selanjutnya ini merupakan tampilan dari “Profil”.

III. KESIMPULAN

Manajemen sesi menggunakan session ID adalah teknik yang efektif dan umum digunakan dalam aplikasi web untuk memastikan autentikasi pengguna berkelanjutan selama sesi berjalan. Diagram ini menunjukkan bahwa setelah login, server membuat sesi yang unik dan mengirim session ID ke browser pengguna. Setiap permintaan berikutnya dari pengguna akan mengandung session ID tersebut, yang memungkinkan server untuk mengenali pengguna dan memberikan respons yang sesuai.

Proses ini tidak hanya memastikan keamanan dalam menjaga sesi pengguna, tetapi juga meningkatkan kenyamanan pengguna dengan menghindari autentikasi ulang yang berulang-ulang dalam satu sesi. Session ID juga dapat diintegrasikan dengan berbagai mekanisme keamanan lainnya untuk mencegah penyalahgunaan, seperti CSRF (Cross-Site Request Forgery) token dan enkripsi session ID.