**Laporan Praktikum**

**Mata Kuliah Pemrograman Web**

A red and black logo

Description automatically generated

**Pertemuan 8.Praktikum7**

**“UTS Studi Kasus Mengenai Sistem Manajemen Kesehatan Mental untuk Pelajar ”**

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**

1. **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.

1. **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.

1. **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.

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

A screenshot of a computer

Description automatically generated

1. Buat database terlebih dahulu pada server xampp “user\_management”

**source code**

CREATE DATABASE mindful;

USE mindful;

CREATE TABLE users (

id INT AUTO\_INCREMENT PRIMARY KEY,

username VARCHAR(100) NOT NULL,

password VARCHAR(255) NOT NULL

);

CREATE TABLE activities (

    id INT AUTO\_INCREMENT PRIMARY KEY,

    name VARCHAR(255) NOT NULL,

    description TEXT NOT NULL,

);

Lalu run hingga muncul ceklis affectrow maka db otomatis telah dibuat

1. Memanggil npm init -y dalam terminal agar node js bisa dijalankan atau node\_modules muncul dalam folder explorer
2. 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

A black text on a white background

Description automatically generated

1. Setelah itu buat folder dengan format struktur seperti ini

A screenshot of a computer

Description automatically generated

1. 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</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

    <style>

        body {

            font-family: Arial, sans-serif;

            background-color: #9ac2eb;

        }

        .container {

            background-color: #fff;

            padding: 30px;

            border-radius: 10px;

            box-shadow: 0 4px 20px rgba(14, 164, 132, 0.242);

            max-width: 400px;

            margin: auto;

            margin-top: 100px;

        }

        h2 {

            text-align: center;

            margin-bottom: 20px;

            color: #1175b3;

        }

        button {

            background-color: #9ac2eb;

            border: none;

            padding: 10px;

        }

        button:hover {

            background-color: #1175b3;

        }

        .link {

            text-align: center;

            margin-top: 20px;

        }

        .link a {

            color: #3789d7;

            text-decoration: none;

        }

        .link a:hover {

            text-decoration: underline;

        }

    </style>

</head>

<body>

    <div class="container">

        <h2>Login</h2>

        <form action="/auth/login" method="POST">

            <div class="mb-3">

                <label for="username" class="form-label">Username</label>

                <input type="text" id="username" name="username" class="form-control" placeholder="Masukkan nama admin" required>

            </div>

            <div class="mb-3">

                <label for="email" class="form-label">Email</label>

                <input type="email" id="email" name="email" class="form-control" placeholder="Masukkan email admin" required>

            </div>

            <div class="mb-3">

                <label for="password" class="form-label">Password</label>

                <input type="password" id="password" name="password" class="form-control" placeholder="Masukkan kata sandi" required>

            </div>

            <button type="submit" class="btn btn-success w-100">Login</button>

        </form>

        <div class="link">

            <p>Don't have an account? <a href="/auth/register">Register Here!</a></p>

        </div>

    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

</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</title>

    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css" rel="stylesheet">

    <style>

        body {

            font-family: Arial, sans-serif;

            background-color: #9ac2eb;

        }

        .container {

            background-color: #fff;

            padding: 30px;

            border-radius: 10px;

            box-shadow: 0 4px 20px rgba(14, 164, 132, 0.242);

            max-width: 400px;

            margin: auto;

            margin-top: 100px;

        }

        h2 {

            text-align: center;

            margin-bottom: 20px;

            color: #1175b3;

        }

        button {

            background-color: #104082;

            border: none;

            padding: 10px;

        }

        button:hover {

            background-color: #77bef1;

        }

        .link {

            text-align: center;

            margin-top: 20px;

        }

        .link a {

            color: #3789d7;

            text-decoration: none;

        }

        .link a:hover {

            text-decoration: underline;

        }

    </style>

</head>

<body>

    <div class="container">

        <h2>Register</h2>

        <form action="/auth/register" method="POST">

            <div class="mb-3">

                <label for="username" class="form-label">Username</label>

                <input type="text" id="username" name="username" class="form-control" placeholder="Masukkan nama admin" required>

            </div>

            <div class="mb-3">

                <label for="email" class="form-label">Email</label>

                <input type="email" id="email" name="email" class="form-control" placeholder="Masukkan email admin" required>

            </div>

            <div class="mb-3">

                <label for="password" class="form-label">Password</label>

                <input type="password" id="password" name="password" class="form-control" placeholder="Masukkan kata sandi" required>

            </div>

            <button type="submit" class="btn btn-success w-100">Register</button>

        </form>

        <div class="link">

            <p>Have an account? <a href="/auth/login">Login Here</a></p>

        </div>

    </div>

    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/js/bootstrap.bundle.min.js"></script>

</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>Mindful Pelajar</title>

    <style>

        /\* Reset CSS \*/

        \* { margin: 0; padding: 0; box-sizing: border-box; }

        body {

            font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

            background-color: #f0f9ff;

            color: #333;

        }

        /\* Header \*/

        header {

            background: linear-gradient(135deg, #0288d1, #00acc1);

            padding: 40px 20px;

            text-align: center;

            color: white;

            border-bottom: 4px solid #00838f;

        }

        header h1 {

            font-size: 3em;

            font-weight: bold;

            margin-bottom: 10px;

        }

        header p {

            font-size: 1.3em;

            max-width: 600px;

            margin: 0 auto 20px;

            line-height: 1.5;

        }

        /\* Navigasi \*/

        nav {

            display: flex;

            justify-content: center;

            background-color: #00acc1;

            padding: 15px 0;

            box-shadow: 0 2px 10px rgba(0, 0, 0, 0.1);

        }

        nav a {

            color: white;

            padding: 10px 20px;

            text-decoration: none;

            font-weight: 600;

            margin: 0 10px;

            border-radius: 20px;

            transition: all 0.3s;

        }

        nav a:hover {

            background-color: #0277bd;

            transform: scale(1.05);

        }

        /\* Konten \*/

        .container {

            max-width: 800px;

            margin: 50px auto;

            padding: 30px;

            background-color: white;

            border-radius: 12px;

            box-shadow: 0 4px 15px rgba(0, 0, 0, 0.15);

            animation: fadeIn 0.5s ease-in-out;

        }

        .container h2 {

            font-size: 2.2em;

            color: #0288d1;

            margin-bottom: 20px;

            border-bottom: 2px solid #00acc1;

            padding-bottom: 5px;

        }

        /\* Gaya untuk section manfaat \*/

        #benefits {

            padding: 20px;

            margin-top: 20px;

            background-color: #e1f5fe;

            border-radius: 8px;

            box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);

        }

        #benefits h2 {

            font-size: 2em;

            color: #0288d1;

            margin-bottom: 15px;

            border-bottom: 2px solid #00acc1;

            padding-bottom: 5px;

        }

        #benefits p {

            font-size: 1.1em;

            margin-bottom: 15px;

            line-height: 1.6;

        }

        #benefits ul {

            padding-left: 20px;

            list-style-type: square;

        }

        #benefits ul li {

            font-size: 1.1em;

            margin: 8px 0;

            line-height: 1.5;

            color: #333;

        }

        /\* Bagian Tips Memulai \*/

        .tips {

            background-color: #e1f5fe;

            padding: 20px;

            border-left: 5px solid #0288d1;

            border-radius: 10px;

            margin-bottom: 30px;

        }

        .tips h3 {

            color: #0288d1;

            margin-bottom: 15px;

            font-size: 1.5em;

            font-weight: bold;

        }

        .tips p {

            font-size: 1.1em;

            margin-bottom: 15px;

        }

        .tips ol {

            list-style-position: inside; /\* Posisi list untuk kesejajaran \*/

            margin-left: 0;

            padding-left: 0;

        }

        .tips li {

            margin-bottom: 10px;

            line-height: 1.6;

        }

        .tips li strong {

            color: #0288d1;

        }

        .container p {

            font-size: 1.1em;

            line-height: 1.8;

            margin-bottom: 20px;

            text-align: justify;

        }

        /\* CRUD section \*/

        .crud-section {

            margin-top: 30px;

            background-color: #e1f5fe;

            padding: 20px;

            border-radius: 8px;

            box-shadow: 0 2px 8px rgba(0, 0, 0, 0.1);

        }

        .crud-section h2 {

            color: #0288d1;

            margin-bottom: 10px;

        }

        .crud-links a {

            display: inline-block;

            padding: 10px 20px;

            margin: 5px;

            background-color: #0288d1;

            color: white;

            text-decoration: none;

            border-radius: 5px;

            transition: background-color 0.3s;

        }

        .crud-links a:hover {

            background-color: #0277bd;

        }

        /\* Footer \*/

        footer {

            text-align: center;

            padding: 20px;

            background-color: #0288d1;

            color: white;

            margin-top: 40px;

            border-radius: 0 0 15px 15px;

        }

        footer p { line-height: 1.5; }

        /\* Responsif \*/

        @media (max-width: 768px) {

            nav { flex-direction: column; padding: 10px; }

            nav a { margin: 5px 0; width: 100%; text-align: center; }

            header h1 { font-size: 2.5em; }

            .container { padding: 20px; }

        }

    </style>

</head>

<body>

    <!-- Header -->

    <header>

        <h1>Mindful Pelajar</h1>

        <p>Menemukan ketenangan dan keseimbangan melalui praktik mindfulness untuk pelajar.</p>

    </header>

    <!-- Navigasi -->

    <nav>

        <a href="#about">Tentang</a>

        <a href="#benefits">Manfaat</a>

        <a href="#tips">Tips Memulai</a>

        <a href="#crud">CRUD Aktivitas </a>

        <a href="/auth/login">Logout</a>

    </nav>

    <!-- Konten Utama -->

    <div class="container">

        <section id="about">

            <h2>Tentang Mindfulness</h2>

            <p>Mindfulness adalah praktik kesadaran penuh yang memungkinkan kita hadir di setiap momen tanpa penilaian. Bagi pelajar, ini sangat penting untuk membantu mengelola stres, meningkatkan fokus, dan memperkuat kesehatan mental. Dengan adopsi mindfulness, pelajar bisa lebih peka terhadap emosi dan kondisi mereka sendiri.</p>

        </section>

        <section id="benefits">

            <h2>Manfaat Mindfulness bagi Pelajar</h2>

            <ul>

                <li>Meningkatkan fokus dan konsentrasi.</li>

                <li>Mengurangi tingkat stres dan kecemasan.</li>

                <li>Meningkatkan kemampuan mengatasi tekanan akademik.</li>

                <li>Memperkuat kesejahteraan emosional dan mental.</li>

            </ul>

        </section>

        <br>

        <section id="tips" class="tips">

            <h3>Tips Memulai Praktik Mindfulness</h3>

            <ol>

                <li><strong>Luangkan waktu:</strong> Dedikasikan 5-10 menit setiap hari untuk meditasi atau sekadar duduk tenang.</li>

                <li><strong>Fokus pada pernapasan:</strong> Amati napas Anda, dan biarkan pikiran lain mengalir tanpa penilaian.</li>

                <li><strong>Praktikkan rasa syukur:</strong> Luangkan waktu untuk merenungkan hal-hal yang Anda syukuri.</li>

                <li><strong>Ikuti kelas atau workshop:</strong> Cari tahu tentang kelas mindfulness di sekolah atau komunitas Anda.</li>

            </ol>

        </section>

 <!-- Section CRUD -->

 <section id="crud" class="CRUD">

    <h2>Kelola Data Aktivitas</h2>

    <p>Lakukan operasi CRUD untuk mengelola aktivitas Anda.</p>

    <div class="crud-links">

        <a href="/add">Tambah Aktivitas</a>

        <a href="/view">Lihat Aktivitas</a>

    </div>

</div>

</div>

    <!-- Footer -->

    <footer>

        <p>&copy; Siti Nurkholizah SIK A UTS 2024 Mindful Pelajar.  Semua hak dilindungi.</p>

    </footer>

</body>

</html>

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

**Source code app.js**

const express = require('express');

const mysql = require('mysql');

const bodyParser = require('body-parser');

const session = require('express-session');

const authRoutes = require('./routes/auth');

const activityRoutes = require('./routes/activity');

const path = require('path');

const Connection = require('./config/db'); // Memastikan koneksi database diimpor dengan benar

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') {

    return res.redirect('/auth/login');

  } else if (req.session.user && req.path === '/') {

    return res.redirect('/auth/profile');

  }

  next();

});

// Routes

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

app.use('/activity', activityRoutes);

// 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');

  }

});

// READ: Tampilkan semua aktivitas

app.get('/view', (req, res) => {

  const query = 'SELECT \* FROM activities';

  Connection.query(query, (err, results) => {

      if (err) {

          console.error("Error executing query:", err);

          res.status(500).send("Internal Server Error");

      } else {

          res.render('view', { activities: results });

      }

  });

});

// CREATE: Tampilkan form untuk menambahkan aktivitas

app.get('/add', (req, res) => {

  res.render('add');

});

// CREATE: Menyimpan aktivitas baru

app.post('/add', (req, res) => {

  const { name, description } = req.body;

  const query = 'INSERT INTO activities (name, description) VALUES (?, ?)';

  Connection.query(query, [name, description], (err, results) => {

      if (err) {

          console.error("Error executing query:", err);

          res.status(500).send("Internal Server Error");

      } else {

          res.redirect('/view');

      }

  });

});

// EDIT: Tampilkan form untuk mengedit aktivitas berdasarkan ID

app.get('/activity/edit/<%= activities.id %>', (req, res) => {

  res.render('edit'); // Pastikan Anda memiliki file edit.ejs di folder views

});

app.post('/activity/edit/<%= activities.id %>', (req, res) => {

  const { name, description } = req.body;

  // Logika untuk memperbarui aktivitas sesuai kebutuhan Anda

  // Misalnya, Anda dapat mencari aktivitas berdasarkan ID dan memperbaruinya di array

  console.log('Nama Aktivitas:', name);

  console.log('Deskripsi:', description);

  // Setelah menyimpan, alihkan ke halaman yang sesuai

  res.redirect('/auth/profile'); // Ganti dengan rute yang sesuai setelah edit

});

// Rute untuk menampilkan daftar aktivitas

app.get('/auth/profile', (req, res) => {

  res.render('profile', { activities }); // Pastikan Anda memiliki file profile.ejs di folder views

});

// DELETE: Menghapus aktivitas berdasarkan ID

app.get('/delete:id', (req, res) => {

  const { id } = req.params;

  const query = 'DELETE FROM activities WHERE id = ?';

  Connection.query(query, [id], (err, results) => {

      if (err) {

          console.error("Error executing query:", err);

          res.status(500).send("Internal Server Error");

      } else {

          res.redirect('/view');

      }

  });

});

app.post('/delete', (req, res) => {

  const { id } = req.params;

  const query = 'DELETE FROM activities WHERE id = ?';

  Connection.query(query, [id], (err, results) => {

      if (err) {

          console.error("Error executing query:", err);

          res.status(500).send("Internal Server Error");

      } else {

          res.redirect('/view');

      }

  });

});

// Menjalankan Server

app.listen(3000, () => {

  console.log('Server running on port 3000');

});

1. Selanjutnya untuk configurasi dengan db.js

**Source code db.js**

const mysql = require('mysql');

const db = mysql.createConnection({

    host: 'localhost',

    user: 'root',

    password: '',

    database: 'mindful'

});

db.connect((err) => {

    if (err) {

        throw err;

    }

    console.log('MySQL connected');

});

module.exports = db;

1. 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 users (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 users 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');

});

// Tambah Aktivitas Baru

router.get('/add', (req, res) => {

    if (req.session.user) {

        res.render('add'); // Render halaman tambah aktivitas

    } else {

        res.redirect('/auth/login');

    }

});

// Edit Aktivitas

router.get('/edit/:id', (req, res) => {

    if (req.session.user) {

        const query = "SELECT \* FROM activities WHERE id = ?";

        db.query(query, [req.params.id], (err, result) => {

            if (err) throw err;

            res.render('edit', { activity: result[0] }); // Render halaman edit aktivitas

        });

    } else {

        res.redirect('/auth/login');

    }

});

// Proses edit aktivitas

router.post('/edit', (req, res) => {

    const { name, description } = req.body;

    const query = "UPDATE activities SET name = ?, description = ? WHERE id = ?";

    db.query(query, [name, description, req.params.id], (err, result) => {

        if (err) throw err;

        res.redirect('/activity/view');

    });

});

// Hapus Aktivitas

router.get('/delete/:id', (req, res) => {

    if (req.session.user) {

        const query = "DELETE FROM activities WHERE id = ?";

        db.query(query, [req.params.id], (err, result) => {

            if (err) throw err;

            res.redirect('/auth/view'); // Redirect ke halaman lihat aktivitas setelah dihapus

        });

    } else {

        res.redirect('/auth/login');

    }

});

// Lihat Aktivitas

router.get('/view', (req, res) => {

    if (req.session.user) {

        const query = "SELECT \* FROM activities WHERE user\_id = ?";

        db.query(query, [req.session.user.id], (err, results) => {

            if (err) throw err;

            res.render('view', { activities: results }); // Render halaman lihat aktivitas

        });

    } else {

        res.redirect('/auth/login');

    }

});

module.exports = router;

**Source Code activity.js**

const express = require('express');

const router = express.Router();

const db = require('../config/db'); // Database configuration

// Lihat semua aktivitas

router.get('/view', (req, res) => {

    const query = "SELECT \* FROM activities WHERE user\_id = ?";

    db.query(query, [req.session.user.id], (err, results) => {

        if (err) throw err;

        res.render('view', { activities: results });

    });

});

// Tambah aktivitas baru - Formulir tambah aktivitas

router.get('/add', (req, res) => {

    res.render('add');

});

// Proses tambah aktivitas baru

router.post('/add', (req, res) => {

    const { name, description } = req.body;

    const query = "INSERT INTO activities (name, description, user\_id) VALUES (?, ?, ?)";

    db.query(query, [name, description, req.session.user.id], (err, result) => {

        if (err) throw err;

        res.redirect('/activity/view');

    });

});

// Edit aktivitas - Formulir edit aktivitas

router.get('/edit', (req, res) => {

    const query = "SELECT \* FROM activities WHERE id = ?";

    db.query(query, [req.params.id], (err, results) => {

        if (err) throw err;

        res.render('edit', { activity: results[0] });

    });

});

// Proses edit aktivitas

router.post('/edit', (req, res) => {

    const { name, description } = req.body;

    const query = "UPDATE activities SET name = ?, description = ? WHERE id = ?";

    db.query(query, [name, description, req.params.id], (err, result) => {

        if (err) throw err;

        res.redirect('/activity/view');

    });

});

// Hapus aktivitas

router.get('/delete/:id', (req, res) => {

    const query = "DELETE FROM activities WHERE id = ?";

    db.query(query, [req.params.id], (err, result) => {

        if (err) throw err;

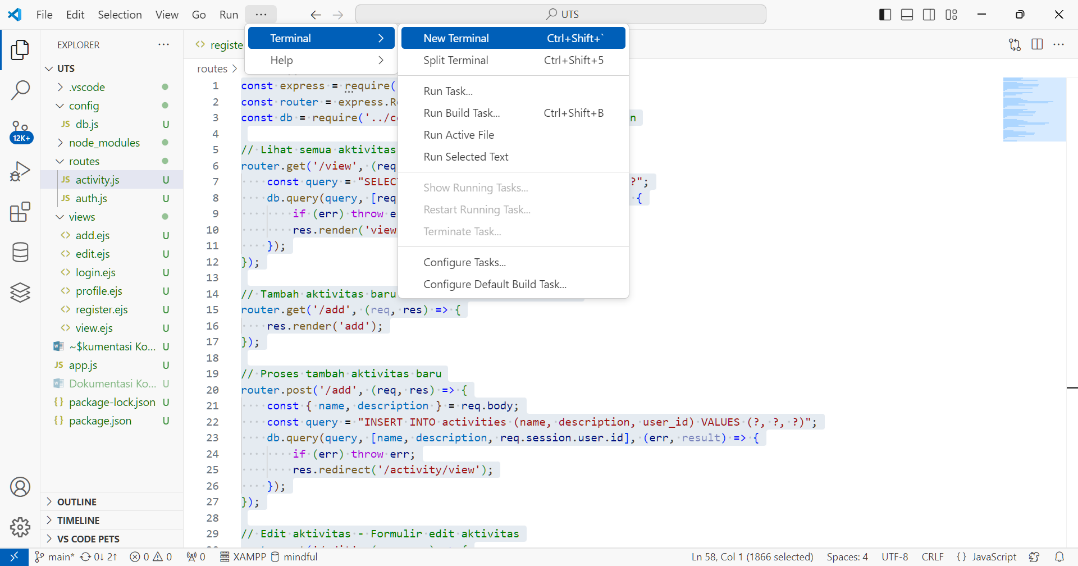
        res.redirect('/activity/view');

    });

});

module.exports = router;

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



A screenshot of a computer

Description automatically generated

Kemudian server akan berjalan diport 3000 dan dapat mengakses halaman web dibrowser melalui URL : http ://localhost:3000.

1. **HASIL DAN PEMBAHASAN**

Setelah server dijalankan, halaman login.ejsakan 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.

**A screenshot of a computer

Description automatically generatedTampilan login.ejs**

**A screenshot of a computer

Description automatically generatedTampilan register.ejs**

A screenshot of a computer

Description automatically generated**Tampilan profile.ejs landpage**

**A screenshot of a computer

Description automatically generatedTampilan bagian informasi manfaat dan tips**

**Tampilan CRUD**

**A screenshot of a computer

Description automatically generated**

**Tambah Aktivitas**

**A screenshot of a computer

Description automatically generated**

**ihat Aktivitas**

**A screenshot of a computer

Description automatically generated**

**Create**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Edit**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Delete**

**A screenshot of a computer

Description automatically generated**

1. **KESIMPULAN**

Dalam praktikum ini, bahwa autentikasi berbasis session dapat diimplementasikan secara efisien menggunakan **Node.js**, **MySQL**, serta dukungan **bcryptjs** untuk enkripsi password dan **express-session** untuk manajemen session. Session menjaga status login pengguna dengan aman selama interaksi dengan aplikasi, memungkinkan akses ke halaman yang dibatasi tanpa memasukkan ulang kredensial. Penggunaan **bcryptjs** melindungi password dengan enkripsi, sedangkan **ejs** memungkinkan perenderan halaman web dinamis sesuai status login. Praktikum ini menunjukkan pemahaman mengenai autentikasi, session, dan enkripsi data dalam aplikasi web yang aman.