Laporan Praktikum

Mata Kuliah

Pemrograman Web

Tugas Pertemuan 5

“CRUD Node JS + MySQL”

Dosen Pengampu:

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

Disusun Oleh:

Habibirrohim

2300149

PROGRAM STUDI SISTEM INFORMASI KELAUTAN

UNIVERSITAS PENDIDIKAN INDONESIA

2024

I. PENDAHULUAN

Praktikum ini bertujuan mengembangkan aplikasi CRUD sederhana menggunakan Node.js dan MySQL untuk sistem manajemen logistik. Fokus utamanya adalah memahami integrasi Node.js dengan database, implementasi operasi CRUD, dan penggunaan Express.js serta EJS untuk pengembangan web full-stack..

II. ALAT DAN BAHAN

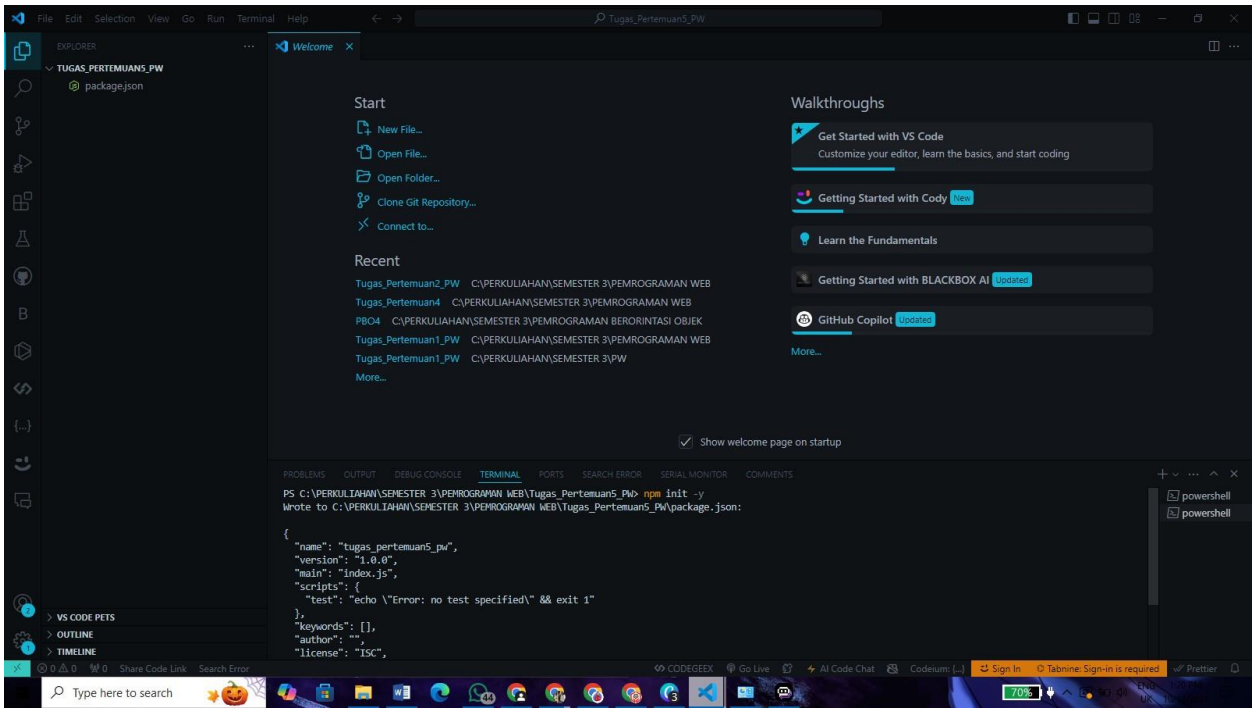
2.1 Alat Dan Bahan

- Node.js
- MySQL
- Express.js
- Body-parser
- EJS (Embedded JavaScript templates)
-Laragon (sebagai server lokal)
- Tailwind CSS (untuk styling)

III. PENJELASAN

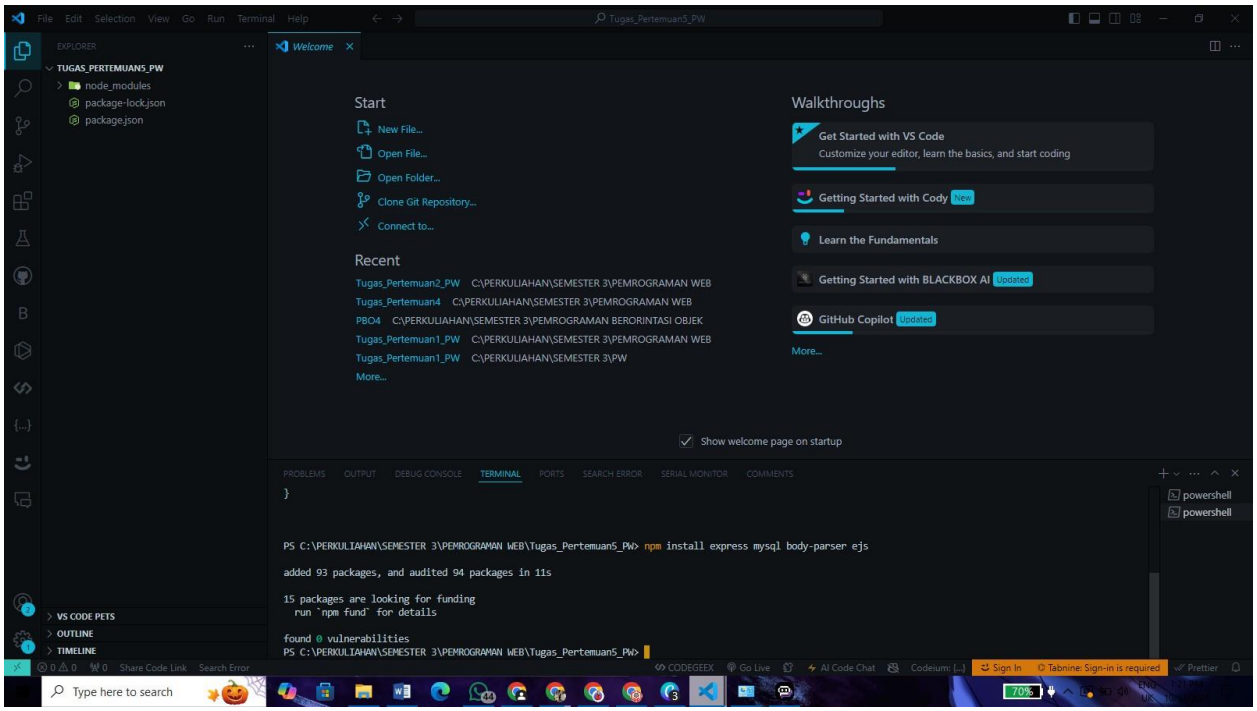
Pertama-tama saya akan membuat study case Logistic ada nama barang, Jumlah berang, dan lokasi barang
Membuat direktori proyek baru: "Tugas_Pertemuan5_PW"
Menginisialisasi proyek Node.js dengan perintah:

Copy
npm init -y
Perintah ini membuat file package.json dengan konfigurasi default.

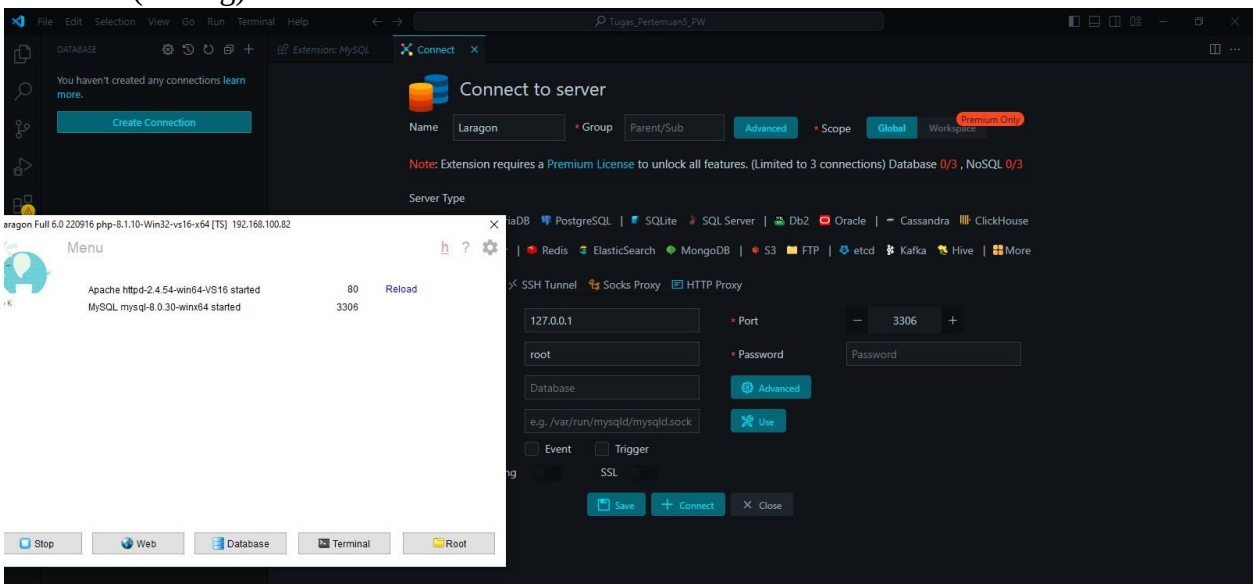


Menginstal paket-paket yang diperlukan dengan menjalankan perintah:

Copy npm install express mysql body-
parser ejs
Proses ini menambahkan 93 paket dan mengaudit 94 paket dalam 11 detik.
1.3 Konfigurasi Database

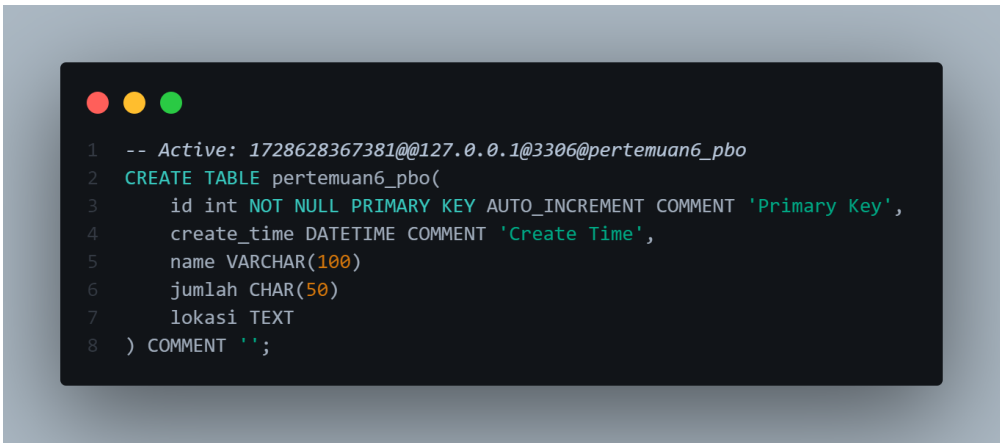


Menggunakan Laragon sebagai server lokal untuk MySQL
MySQL berjalan pada port 3306 Menggunakan kredensial default:
Host: 127.0.0.1
Username: root
Password: (kosong)



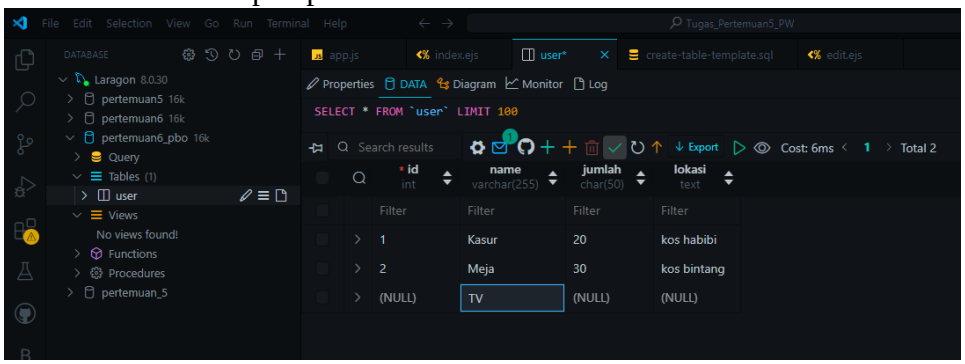
Membuat database baru bernama "pertemuan6_pbo"
Membuat tabel 'user' dengan kolom:

id (auto-increment)
name
jumlah
lokasi
[Penjelasan untuk gambar konfigurasi database]
Gambar ini menunjukkan proses pembuatan tabel 'user' di phpMyAdmin. Kolom-kolom diatur sesuai dengan kebutuhan aplikasi, dengan 'id' sebagai primary key yang auto-increment.



```sql
-- Active: 1728628367381@@127.0.0.1@3306@pertemuan6_pbo
CREATE TABLE pertemuan6_pbo(
    id int NOT NULL PRIMARY KEY AUTO_INCREMENT COMMENT 'Primary Key',
    create_time DATETIME COMMENT 'Create Time',
    name VARCHAR(100)
    jumlah CHAR(50)
    lokasi TEXT
) COMMENT '';
```

Crate input table dengan klik tanda + sesuai data yang diperlukan
Setelah itu klik simpan pada tanda ceklis



Impor modul yang diperlukan:
express: Framework web untuk Node.js
mysql: Modul untuk koneksi ke database MySQL
body-parser: Middleware untuk parsing body request
ejs: Template engine untuk rendering view
Inisialisasi aplikasi Express dan konfigurasi middleware:
Menggunakan body-parser untuk parsing URL-encoded dan JSON
Konfigurasi koneksi database MySQL:
Host: localhost
User: root
Password: (kosong)

Database: pertemuan6_pbo
Menangani koneksi database:
Menampilkan pesan error jika gagal terkoneksi
Menampilkan pesan sukses beserta thread ID jika berhasil terkoneksi
Konfigurasi view engine menggunakan EJS
Implementasi route untuk READ operation:
Mengambil semua data dari tabel 'user'
Merender view 'index' dengan data yang diambil
Menjalankan server pada port 3000

```
const express = require (`express`);
const mysql = require(`mysql`);
const bodyParser = require(`body-parser`);
const { name } = require("ejs");

const app = express();
app.use(bodyParser.urlencoded({extended : false}));
app.use(bodyParser.json());

const connection = mysql.createConnection({
    host:`localhost`,
    user: `root`,
    password: ``,
    database: `pertemuan6_pbo`
})

connection.connect((err)=>{
    if(err) {
        console.error("eror connecting to Mysql:", err.stack);
        return;
    }
    console.log("Mysql Connected" + connection.threadId);
});

app.set(`view engine`, `ejs`);
//ini adalaha routing (create, read, update, delete)
//READ
app.get(`/`, (req, res) => {
    const query = `SELECT * FROM user`;
    connection.query(query, (err, result) =>{
        res.render(`index`, {user: result});
    });
});

app.listen(3000, () => {
    console.log(" running on port 3000, Buka web melalui http://localhost:3000");
})
```

CREATE Operation:
Route: POST '/add'
Mengambil data dari body request
Menyisipkan data baru ke dalam tabel 'user'

Redirect ke halaman utama setelah penyisipan berhasil
UPDATE Operation:
Route GET '/edit/:id': Untuk mengakses halaman edit
Mengambil data user berdasarkan ID
Merender view 'edit' dengan data user yang akan diedit
Route POST '/update/:id': Untuk memproses update
Mengupdate data user di database
Redirect ke halaman utama setelah update berhasil
DELETE Operation:
Route: GET '/delete/:id'
Menghapus data user berdasarkan ID
Redirect ke halaman utama setelah penghapusan berhasil

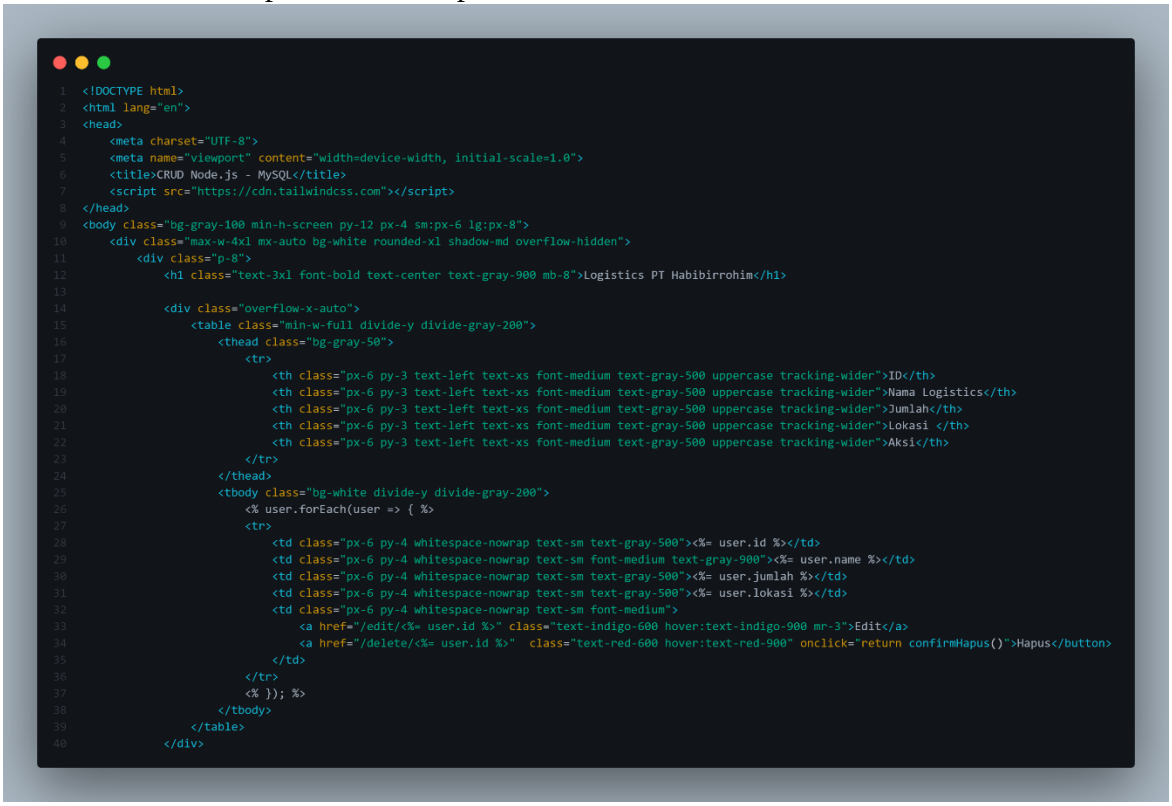
```
//CREATE /INPUT/INSERT
app.post('/add', (req, res) => {
    const { name, jumlah, lokasi } = req.body;
    const query = `INSERT INTO user (name, jumlah, lokasi) VALUES (?, ?, ?)`;
    connection.query(query, [name, jumlah, lokasi], (err, result) => {
      if (err) throw err;
      res.redirect('/');
    });
  });

//UPDATE
//untuk akses halaman
app.get('/edit/:id', (req, res) => {
    const query = `SELECT * FROM user WHERE id = ?`;
    connection.query(query, [req.params.id], (err, result) => {
        if(err) throw err;
        res.render('edit', { user: result[0] });
    });
});

//untuk update data
app.post('/update/:id', (req, res) => {
    const { name, jumlah, lokasi } = req.body;
    const query = `UPDATE user SET name = ?, jumlah = ?, lokasi = ? WHERE id = ?`;
    connection.query(query, [name, jumlah, lokasi, req.params.id], (err, result) => {
      if (err) throw err;
      res.redirect('/');
    });
  });

//hapus
app.get('/delete/:id', (req, res) => {
    const query = `DELETE FROM user WHERE id = ?`;
    connection.query(query, [req.params.id], (err, result) => {
      if (err) throw err;
      res.redirect('/');
    });
  });
```

Lalu di file index.ejs saya menggunakan framework Tailwindcss
Penggunaan Tailwind CSS untuk styling
Judul halaman: "Logistics PT Habibirrohim"
Tabel untuk menampilkan data user:
Kolom: ID, Nama Logistics, Jumlah, Lokasi, Aksi
Menggunakan loop untuk menampilkan semua data user
Tombol Edit dan Hapus untuk setiap baris data

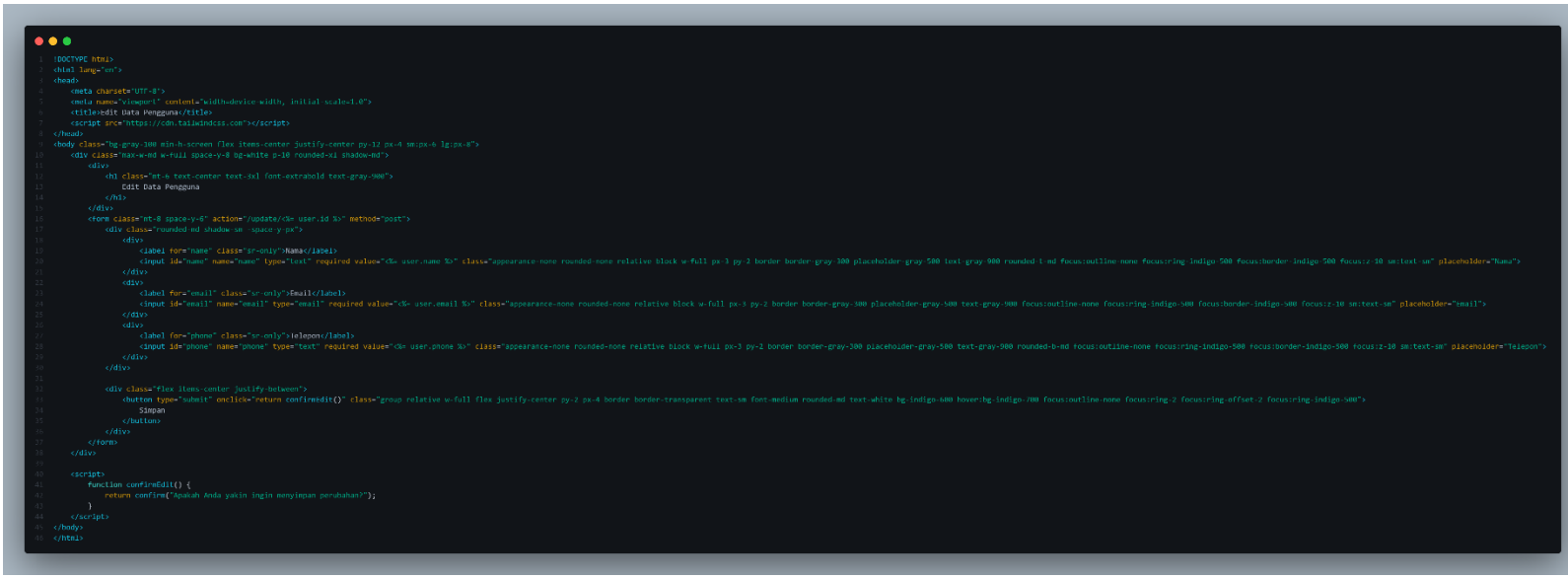


```ejs
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>CRUD Node.js - MySQL</title>
    <script src="https://cdn.tailwindcss.com"></script>
</head>
<body class="bg-gray-100 min-h-screen py-12 px-4 sm:px-6 lg:px-8">
    <div class="max-w-4xl mx-auto bg-white rounded-xl shadow-md overflow-hidden">
        <div class="p-8">
            <h1 class="text-3xl font-bold text-center text-gray-900 mb-8">Logistics PT Habibirrohim</h1>

            <div class="overflow-x-auto">
                <table class="min-w-full divide-y divide-gray-200">
                    <thead class="bg-gray-50">
                        <tr>
                            <th class="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">ID</th>
                            <th class="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Nama Logistics</th>
                            <th class="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Jumlah</th>
                            <th class="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Lokasi </th>
                            <th class="px-6 py-3 text-left text-xs font-medium text-gray-500 uppercase tracking-wider">Aksi</th>
                        </tr>
                    </thead>
                    <tbody class="bg-white divide-y divide-gray-200">
                        <% user.forEach(user => { %>
                        <tr>
                            <td class="px-6 py-4 whitespace-nowrap text-sm text-gray-500"><%= user.id %></td>
                            <td class="px-6 py-4 whitespace-nowrap text-sm font-medium text-gray-900"><%= user.name %></td>
                            <td class="px-6 py-4 whitespace-nowrap text-sm text-gray-500"><%= user.jumlah %></td>
                            <td class="px-6 py-4 whitespace-nowrap text-sm text-gray-500"><%= user.lokasi %></td>
                            <td class="px-6 py-4 whitespace-nowrap text-sm font-medium">
                                <a href="/edit/<%= user.id %>" class="text-indigo-600 hover:text-indigo-900 mr-3">Edit</a>
                                <a href="/delete/<%= user.id %>"  class="text-red-600 hover:text-red-900" onclick="return confirmHapus()">Hapus</button>
                            </td>
                        </tr>
                        <% }); %>
                    </tbody>
                </table>
            </div>
```

```ejs
            <div class="mt-12">
                <h2 class="text-2xl font-bold text-gray-900 mb-6">Tambah Logistics</h2>
                <form action="/add" method="post" class="space-y-6">
                    <div>
                        <label for="name" class="block text-sm font-medium text-gray-700">Nama Logistics:</label>
                        <input type="text" id="name" name="name" required class="mt-1 block w-full border border-gray-300 rounded-md shadow-sm py-2 px-3 focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm">
                    </div>
                    <div>
                        <label for="jumlah" class="block text-sm font-medium text-gray-700">Jumlah:</label>
                        <input type="jumlah" id="jumlah" name="jumlah" required class="mt-1 block w-full border border-gray-300 rounded-md shadow-sm py-2 px-3 focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm">
                    </div>
                    <div>
                        <label for="lokasi" class="block text-sm font-medium text-gray-700">Lokasi:</label>
                        <input type="lokasi" id="lokasi" name="lokasi" required class="mt-1 block w-full border border-gray-300 rounded-md shadow-sm py-2 px-3 focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 sm:text-sm">
                    </div>
                    <div>
                        <button type="submit" onclick="return confirmEdit()"class="w-full flex justify-center py-2 px-4 border border-transparent rounded-md shadow-sm text-sm font-medium text-white bg-indigo-600 hover:bg-indigo-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500">
                            Tambah
                        </button>
                    </div>
                </form>
            </div>
        </div>
    </div>
</body>
<script>
    function confirmHapus (){
        return confirm("Apakah Anda Yakin Akan Menghapus Data ini ?");
    }
    function confirmEdit (){
        return confirm("Data Berhasil Ditambahkan");
    }
</script>
</html>
```

Judul form: "Edit Data Pengguna"
Form action mengarah ke route update dengan ID user
Input fields yang sudah terisi dengan data user yang akan diedit:
Nama
Email
Telepon
Tombol Submit untuk menyimpan perubahan
Script JavaScript untuk konfirmasi edit data



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Edit Data Pengguna</title>
    <script src="https://cdn.tailwindcss.com"></script>
</head>
<body class="bg-gray-100 min-h-screen flex items-center justify-center py-12 px-4 sm:px-6 lg:px-8">
    <div class="max-w-md w-full space-y-8 bg-white p-10 rounded-xl shadow-md">
        <div>
            <h1 class="mt-6 text-center text-3xl font-extrabold text-gray-900">
                Edit Data Pengguna
            </h1>
        </div>
        <form class="mt-8 space-y-6" action="/update/<%= user.id %>" method="post">
            <div class="rounded-md shadow-sm -space-y-px">
                <div>
                    <label for="name" class="sr-only">Nama</label>
                    <input id="name" name="name" type="text" required value="<%= user.name %>" class="appearance-none rounded-none relative block w-full px-3 py-2 border border-gray-300 placeholder-gray-500 text-gray-900 rounded-t-md focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 focus:z-10 sm:text-sm" placeholder="Nama">
                </div>
                <div>
                    <label for="email" class="sr-only">Email</label>
                    <input id="email" name="email" type="email" required value="<%= user.email %>" class="appearance-none rounded-none relative block w-full px-3 py-2 border border-gray-300 placeholder-gray-500 text-gray-900 focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 focus:z-10 sm:text-sm" placeholder="email">
                </div>
                <div>
                    <label for="phone" class="sr-only">Telepon</label>
                    <input id="phone" name="phone" type="text" required value="<%= user.phone %>" class="appearance-none rounded-none relative block w-full px-3 py-2 border border-gray-300 placeholder-gray-500 text-gray-900 rounded-b-md focus:outline-none focus:ring-indigo-500 focus:border-indigo-500 focus:z-10 sm:text-sm" placeholder="Telepon">
                </div>
            </div>

            <div class="flex items-center justify-between">
                <button type="submit" onclick="return confirmEdit()" class="group relative w-full flex justify-center py-2 px-4 border border-transparent text-sm font-medium rounded-md text-white bg-indigo-600 hover:bg-indigo-700 focus:outline-none focus:ring-2 focus:ring-offset-2 focus:ring-indigo-500">
                    Simpan
                </button>
            </div>
        </form>
    </div>

    <script>
        function confirmEdit() {
            return confirm("Apakah Anda yakin ingin menyimpan perubahan?");
        }
    </script>
</body>
</html>
```

IV. HASIL PEMBAHASAN

Dalam praktikum ini, mahasiswa berhasil mengimplementasikan aplikasi manajemen logistik. Proses dimulai dengan persiapan lingkungan pengembangan, termasuk inisialisasi proyek Node.js dan instalasi dependensi. Selanjutnya, konfigurasi database MySQL dilakukan menggunakan Laragon sebagai server lokal. Backend diimplementasikan dengan Express.js, mencakup rute untuk operasi CRUD. Frontend dikembangkan menggunakan EJS sebagai template engine dan Tailwind CSS untuk styling. Aplikasi yang dihasilkan memiliki fitur tampilan tabel data logistik, fungsi tambah, edit, dan hapus data, serta form untuk mengedit entri yang ada.

V. KESIMPULAN

Praktikum ini berhasil memberikan pengalaman langsung dalam pengembangan aplikasi web full-stack. Mahasiswa memperoleh pemahaman tentang integrasi Node.js dengan MySQL, implementasi CRUD, dan penggunaan framework modern untuk pengembangan web. Keterampilan yang diperoleh memberi fondasi kuat untuk proyek pengembangan web yang lebih kompleks di masa depan.