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 untuk membuat aplikasi CRUD (Create, Read, Update, Delete) sederhana menggunakan Node.js sebagai backend dan MySQL sebagai database. Aplikasi ini akan mengelola data pengguna dengan informasi nama, email, dan nomor telepon.

II. ALAT DAN BAHAN

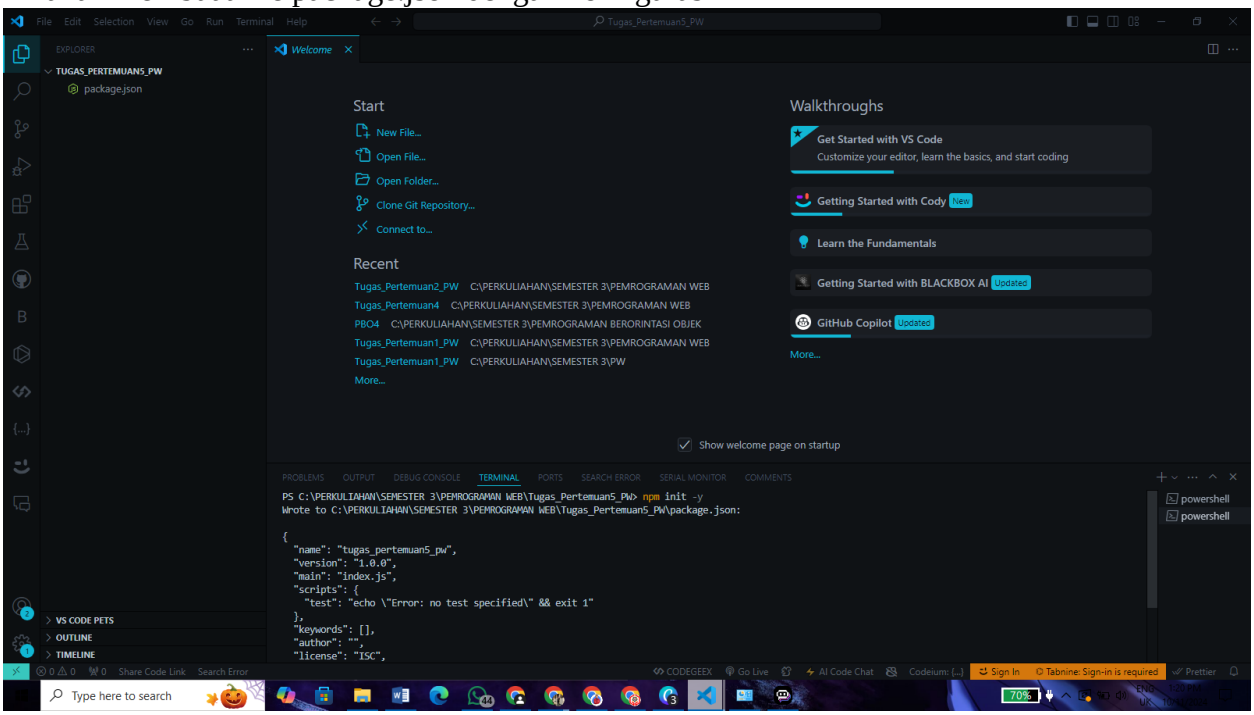
2.1 Alat Dan Bahan

- Node.js
- MySQL
- Express.js
- Body-parser
- EJS (Embedded JavaScript templates)

III. PENJELASAN

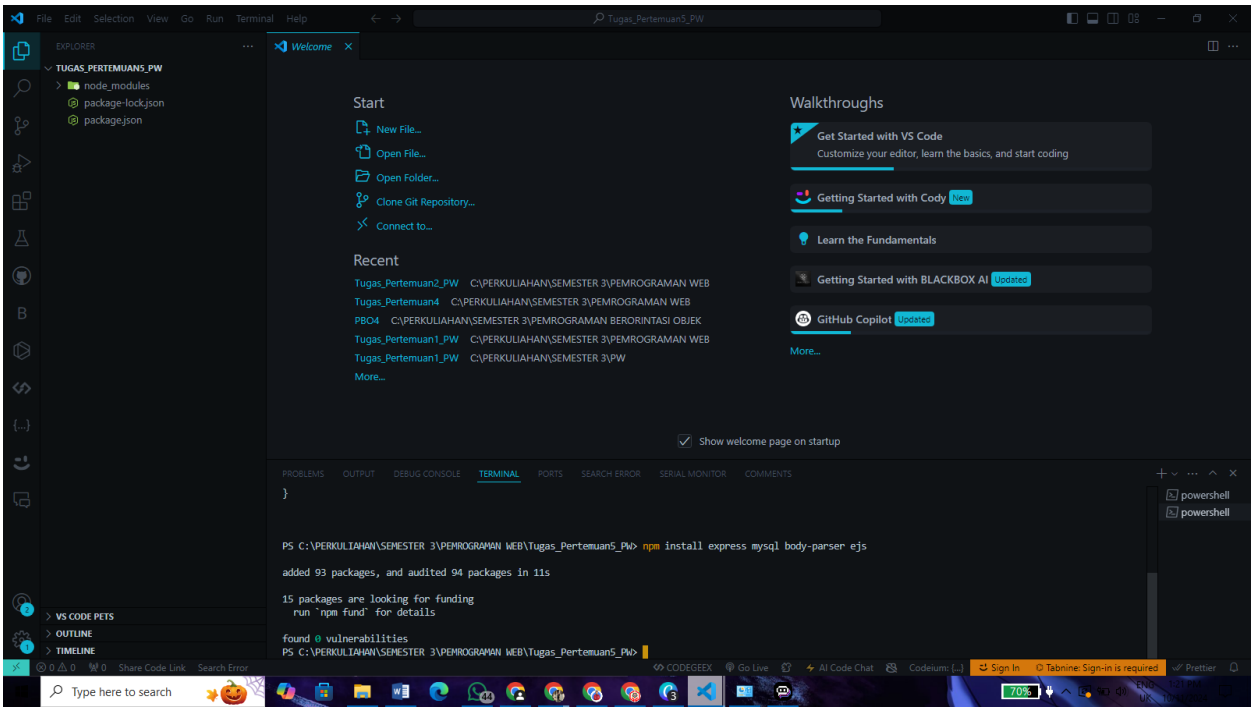
Membuat direktori proyek baru: "Tugas_Pertemuan5_PW"
Menginisialisasi proyek Node.js dengan menjalankan perintah:

Copy
npm init -y
Ini akan membuat file package.json dengan konfigurasi

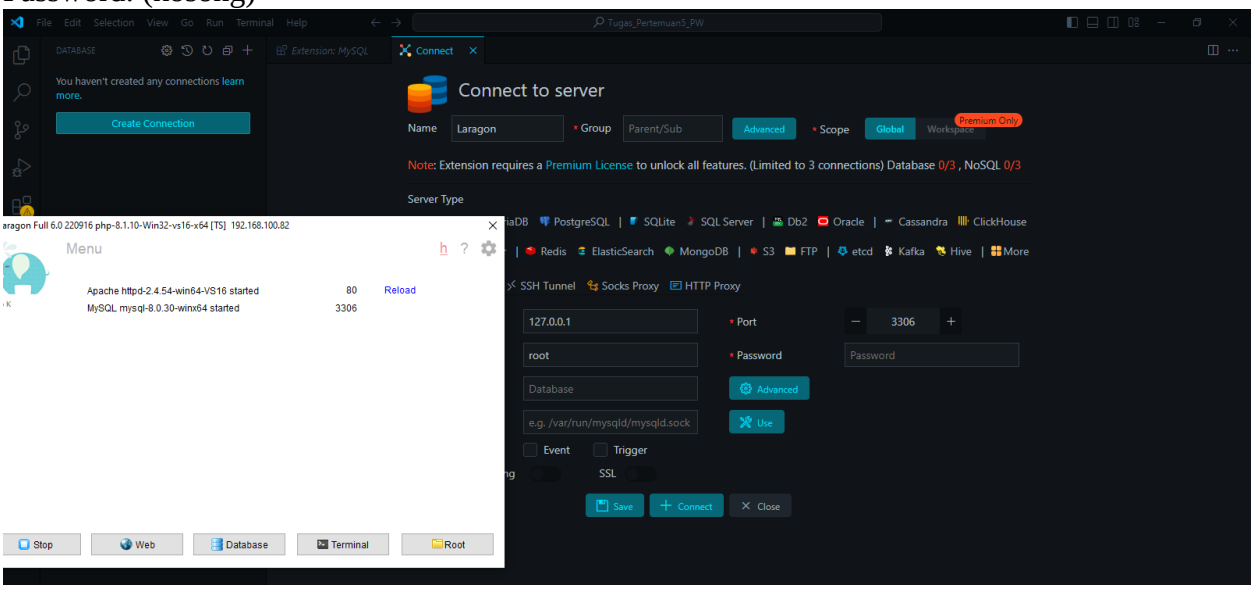


Menginstal paket-paket yang diperlukan dengan menjalankan perintah:

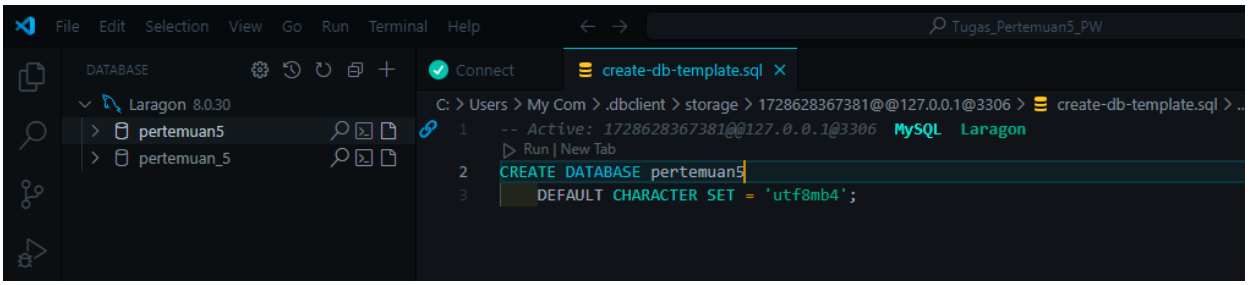
Copy
npm install express mysql body-parser ejs
Proses ini menambahkan 93 paket dan mengaudit 94 paket dalam 11 detik.
1.3 Konfigurasi Database



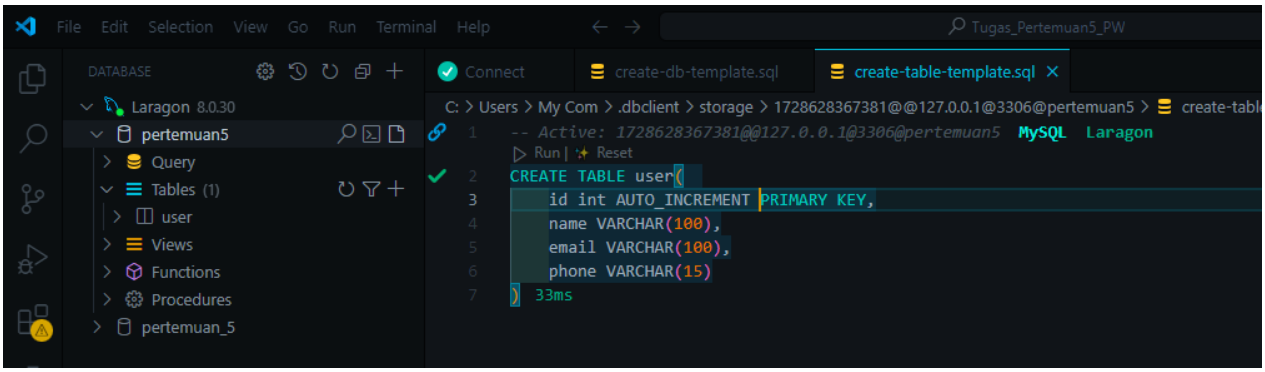
Menggunakan Laragon sebagai server lokal untuk MySQL
MySQL berjalan pada port 3306
Menggunakan kredensial default:
Host: 127.0.0.1
Username: root
Password: (kosong)



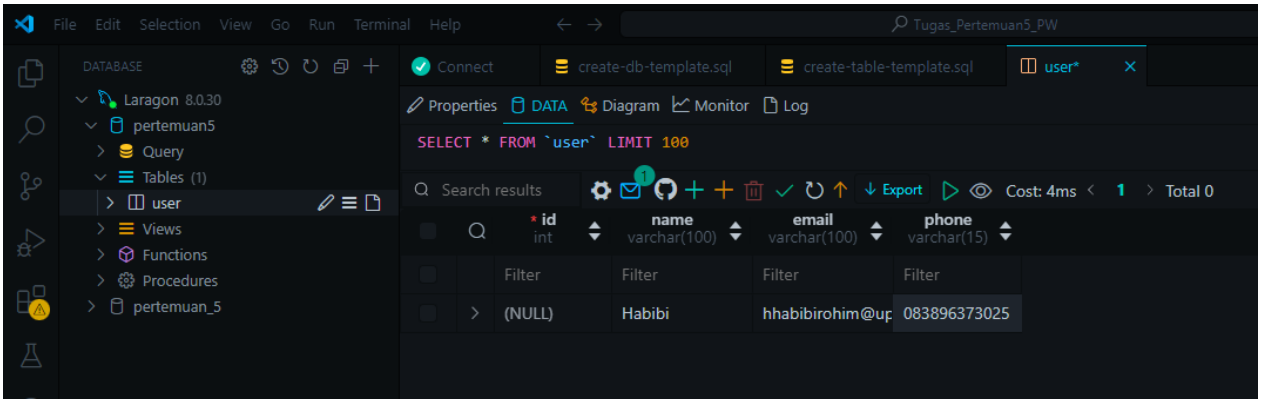
Membuat database baru bernama "pertemuan5" (sesuai dengan kode dibawah)



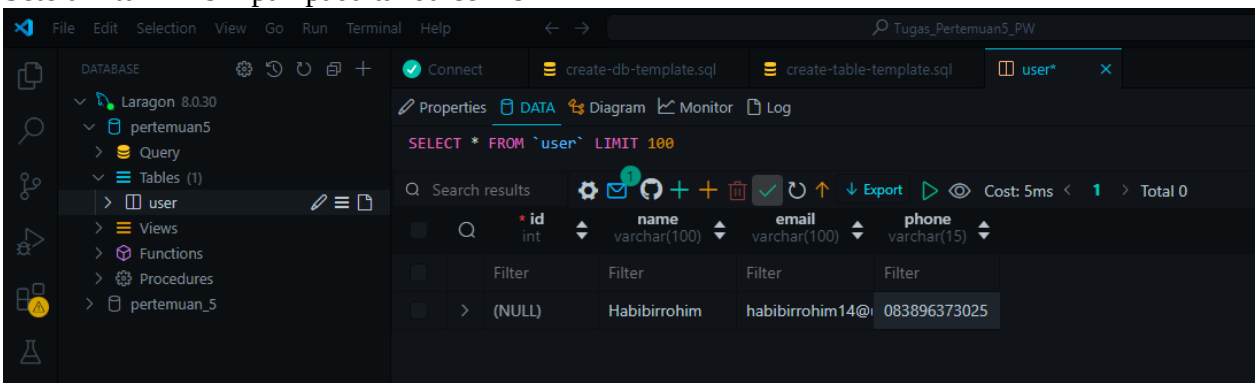
Membuat Table database dengan menambahkan id name, email, phone sesuaidengan dibawah



Crate input table dengan klik tanda + sesuai data yang diperlukan

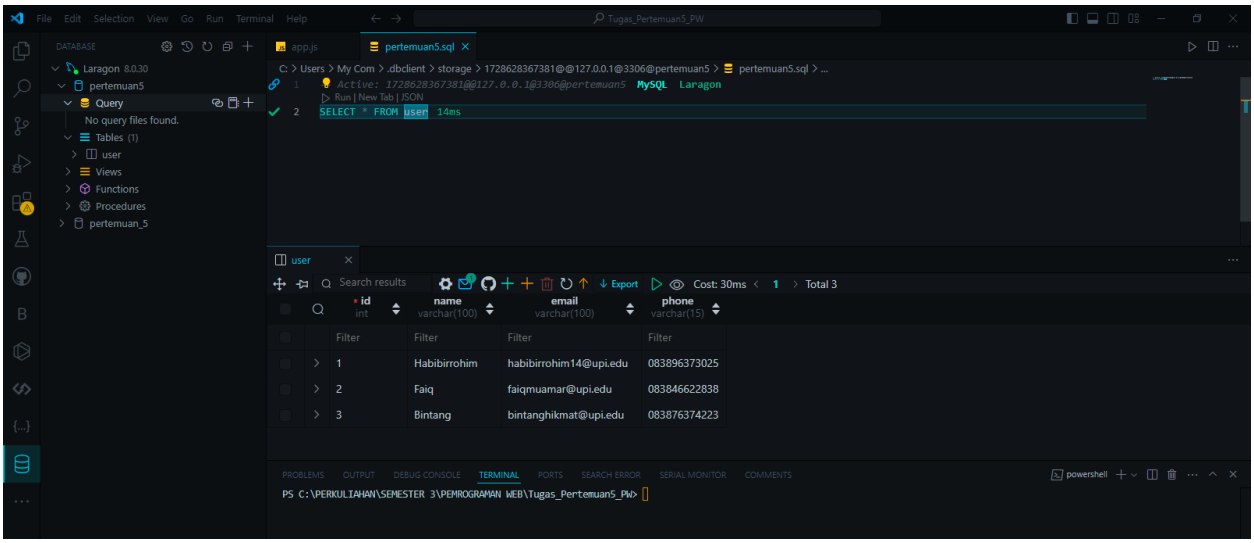


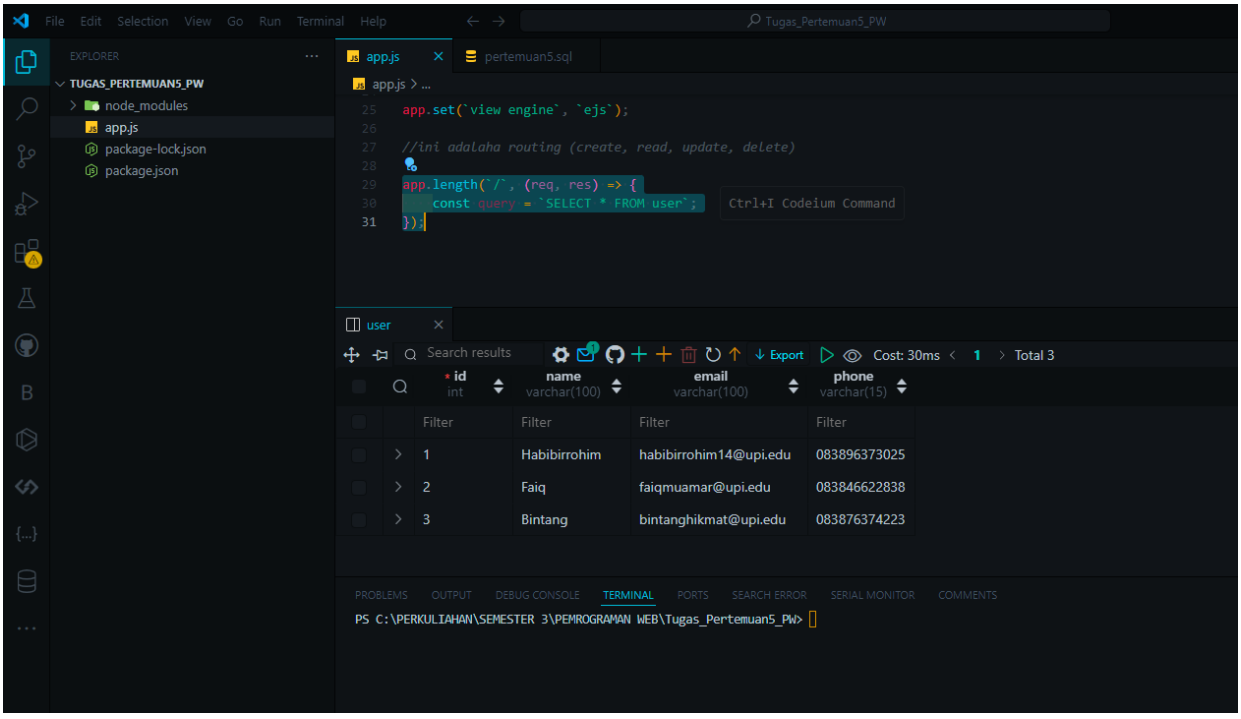
Setelah itu klik simpan pada tanda ceklis



CRUD

a. Read (Membaca Data)
- Membuat route GET '/' untuk menampilkan semua data pengguna
- Menggunakan query SQL SELECT untuk mengambil data dari tabel user





Mengimpor modul yang diperlukan (express, mysql, body-parser, ejs).
Membuat instance Express dan mengkonfigurasi middleware.
Membuat koneksi ke database MySQL.
Mengatur EJS sebagai view engine.

```
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: `pertemuan5`
})

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`);
```

CRUD (Create, Read, Update, Delete): a. Read (Membaca Data):
Menggunakan route GET '/' untuk menampilkan semua data pengguna.
Mengeksekusi query SELECT untuk mengambil semua data dari tabel 'user'.
Merender halaman 'index' dengan data pengguna yang diambil.

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

```

Create (Menambah Data):
Menggunakan route POST '/add' untuk menambah pengguna baru.
Mengekstrak data dari body request.
Mengeksekusi query INSERT untuk menambahkan data ke tabel 'user'.
Melakukan redirect ke halaman utama setelah penambahan berhasil.
Update (Mengubah Data):
Menggunakan route GET '/edit/:id' untuk menampilkan form edit.
Mengambil data pengguna berdasarkan ID untuk ditampilkan di form.
Menggunakan route POST '/update/:id' untuk memproses perubahan data.
Mengeksekusi query UPDATE untuk mengubah data di tabel 'user'.
Melakukan redirect ke halaman utama setelah perubahan berhasil.
Delete (Menghapus Data):
Menggunakan route GET '/delete/:id' untuk menghapus data pengguna.
Mengeksekusi query DELETE untuk menghapus data dari tabel 'user'.
Melakukan redirect ke halaman utama setelah penghapusan berhasil.

```
//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, email, phone } = req.body;
    const query = `UPDATE user SET name = ?, email = ?, phone = ? WHERE id = ?`;
    connection.query(query, [name, email, phone, 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('/');
    });
  });
```

Tampilan (Views):

index.ejs: Menampilkan daftar pengguna dalam bentuk tabel dan form untuk menambah pengguna baru.

edit.ejs: Menampilkan form untuk mengedit data pengguna yang sudah ada.

Penjelasan Kode HTML:

index.ejs:

Menampilkan judul dan tabel daftar pengguna.

Menggunakan loop untuk menampilkan setiap pengguna dari database.

Menyediakan link untuk edit dan hapus untuk setiap pengguna.

Menampilkan form untuk menambah pengguna baru.

```html
<!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>
</head>
<body>
    <h1>Daftar User/Pengguna</h1>
    <table border="1">
        <tr>
            <th>ID</th>
            <th>Name</th>
            <th>Email</th>
            <th>Telepon</th>
            <th>Aksi</th>
        </tr>
        <% user.forEach(user => { %>
        <tr>
            <td><%= user.id %></td>
            <td><%= user.name %></td>
            <td><%= user.email %></td>
            <td><%= user.phone %></td>
            <td>
                <a href="/edit/<%= user.id %>">Edit</a>
                <a href="/delete/<%= user.id %>">Hapus</a>
            </td>
        </tr>
        <% }); %>
    </table>

    <h2>Tambah Pengguna Baru </h2>
    <form action="/add" method="post">
        <label for="name">Name:</label>
        <input type="text" id="name" name="name"required>
        <br>
        <label for="email">Email:</label>
        <input type="email" id="email" name="email" required><br>
        <label for="phone">Telepon:</label>
        <input type="text" id="phone" name="phone" required><br>
        <button type="submit">Tambah</button>
    </form>
</body>
</html>
```

edit.ejs:
Menampilkan form untuk mengedit data pengguna.
Mengisi nilai awal form dengan data pengguna yang sedang diedit.

```html
<!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>
</head>
<body>
    <h1>Edit Data Pengguna</h1>
    <form action="/update/<%= user.id %>" method="post">
        <label for="name">name:</label>
        <input type="text" id="name" name="name" required value="<%= user.name %>">
        <br>
        <label for="email">email:</label>
        <input type="email" id="email" name="email" required value="<%= user.email %>">
        <br>
        <label for="phone">telepon:</label>
        <input type="text" id="phone" name="phone" required value="<%= user.phone %>">
        <br>
        <button type="submit">Simpan</button>
    </form>
</body>
</html>
```

Hasil dan Pembahasan

Aplikasi CRUD berhasil dibuat dengan fitur-fitur berikut:

- Menampilkan daftar pengguna dalam bentuk tabel
- Menambahkan pengguna baru melalui form
- Mengedit data pengguna yang sudah ada
- Menghapus data pengguna

V. KESIMPULAN

Praktikum ini berhasil mengimplementasikan operasi CRUD dasar menggunakan Node.js dan MySQL. Mahasiswa dapat memahami cara kerja backend sederhana dan interaksinya dengan database relasional.