

Laporan Praktikum
Mata Kuliah Pemrograman WEB



PERTEMUAN 5. TUGAS 1
CREATE, READ, UPDATE DAN DELETE

Dosen Pengampu:
Wildan Aprizall Arifin, S.Pd., M.Kom

Disusun Oleh:
Nelita Maharani
2307052

PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA

2024

I. PENDAHULUAN

CRUD merupakan konsep fundamental dalam dunia pengembangan aplikasi yang mengacu pada empat operasi dasar dalam pengelolaan data: Create (membuat), Read (membaca), Update (memperbarui), dan Delete (menghapus). Konsep ini menjadi pondasi penting dalam hampir setiap aplikasi yang melibatkan manipulasi data, mulai dari aplikasi sederhana hingga sistem yang kompleks.

Dalam konteks pengembangan aplikasi, operasi Create memungkinkan pengguna untuk menambahkan data baru ke dalam sistem. Misalnya, ketika seseorang mendaftar di sebuah platform, data mereka "dibuat" dan disimpan dalam database. Operasi Read digunakan untuk mengambil dan menampilkan data yang telah tersimpan, seperti ketika aplikasi menampilkan daftar pengguna atau detail produk. Update memungkinkan modifikasi data yang sudah ada, contohnya ketika pengguna mengubah profil mereka atau admin memperbarui harga produk. Sementara itu, Delete digunakan untuk menghapus data yang tidak diperlukan lagi dari sistem.

Implementasi CRUD dapat dilakukan melalui berbagai cara, tergantung pada teknologi yang digunakan. Dalam konteks database SQL, operasi Create dilakukan menggunakan perintah INSERT, Read menggunakan SELECT, Update dengan UPDATE, dan Delete menggunakan DELETE. Pada pengembangan API RESTful, operasi-operasi ini umumnya dipetakan ke metode HTTP: POST untuk Create, GET untuk Read, PUT atau PATCH untuk Update, dan DELETE untuk operasi penghapusan.

Ketika mengimplementasikan CRUD, ada beberapa praktik terbaik yang perlu diperhatikan. Pertama, validasi data sangat penting untuk memastikan integritas data yang disimpan. Kedua, aspek keamanan harus diutamakan, termasuk implementasi autentikasi dan otorisasi untuk mencegah akses tidak sah ke operasi CRUD. Ketiga, memberikan umpan balik yang jelas kepada pengguna setelah setiap operasi CRUD sangat penting untuk pengalaman pengguna yang baik. Terakhir, optimasi performa, terutama untuk operasi yang melibatkan data dalam jumlah besar, tidak boleh diabaikan.

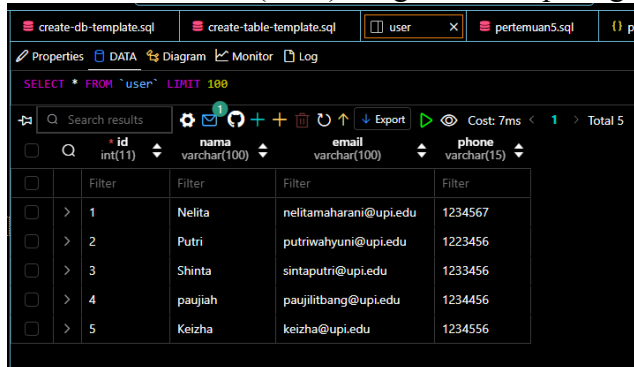
Sebagai contoh implementasi, dalam pengembangan aplikasi web menggunakan Node.js dan MySQL, operasi CRUD dapat direalisasikan menggunakan query database yang sesuai. Untuk operasi Create, aplikasi akan menggunakan query INSERT untuk menambahkan data baru. Operasi Read menggunakan SELECT untuk mengambil data yang diperlukan. Update dilakukan dengan query UPDATE untuk memodifikasi data yang sudah ada, dan Delete menggunakan query DELETE untuk menghapus data.

II. ALAT DAN BAHAN

1. Komputer/Laptop
2. Visual Studio Code
3. XAMPP
4. MySQL
5. Chrome

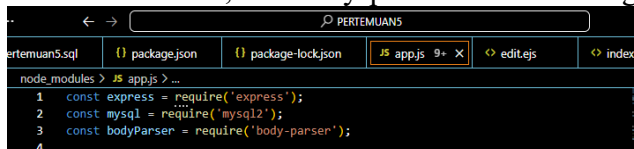
III. LANGKAH KERJA

1. Membuat database (table) dengan hasil seperti gambar dibawah ini.



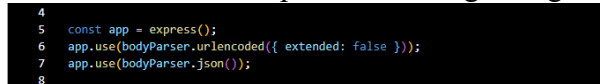
id	nama	email	phone
1	Nelita	nelitamaharani@upi.edu	1234567
2	Putri	putriwahyuni@upi.edu	1223456
3	Shinta	sintaputri@upi.edu	1233456
4	paujiah	paujilitbang@upi.edu	1234456
5	Keizha	keizha@upi.edu	1234556

2. Membuat kode CRUD menggunakan Node.js, Express dan SQL. Setelahnya, impor modul yang diperlukan, seperti Express untuk server web, mysql2 untuk koneksi database, dan body-parser untuk menangani data dari form.



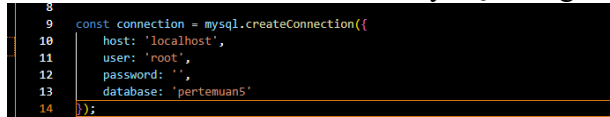
```
1 const express = require('express');
2 const mysql = require('mysql2');
3 const bodyParser = require('body-parser');
4
```

3. Membuat instance express dan mengkonfigurasi middleware body-parser.



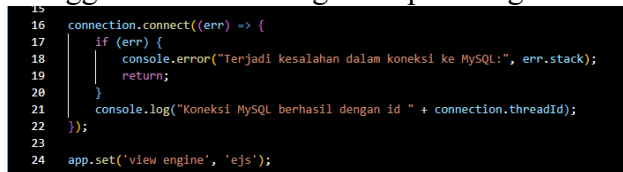
```
4
5 const app = express();
6 app.use(bodyParser.urlencoded({ extended: false }));
7 app.use(bodyParser.json());
8
```

4. Membuat koneksi ke database MySQL dengan parameter yang sesuai.



```
8
9 const connection = mysql.createConnection({
10   host: 'localhost',
11   user: 'root',
12   password: '',
13   database: 'pertemuan5'
14 });
15
```

5. Menggunakan EJS sebagai template engine untuk tampilan.



```
15
16 connection.connect((err) => {
17   if (err) {
18     console.error("Terjadi kesalahan dalam koneksi ke MySQL:", err.stack);
19     return;
20   }
21   console.log("Koneksi MySQL berhasil dengan id " + connection.threadId);
22 });
23
24 app.set('view engine', 'ejs');
25
```

6. Mengambil semua data dari tabel users dan menampilkannya di halaman utama.



```
28 // Read
29 app.get('/', (req, res) => {
30   const query = "SELECT * FROM users";
31   connection.query(query, (err, result) => {
32     if (err) {
33       console.error("Terjadi kesalahan dalam koneksi ke MySQL:", err.stack);
34       res.status(500).send("Terjadi kesalahan pada server");
35       return;
36     }
37     res.render("index", { users: result });
38   });
39 });
40
```

7. Menambahkan data baru ke database.

```
40
41 // Create / Input / Insert
42 app.post('/add', (req, res) => {
43   const { name, email, phone } = req.body;
44   const query = 'INSERT INTO users (name, email, phone) VALUES (?, ?, ?)';
45   connection.query(query, [name, email, phone], (err, result) => {
46     if (err) throw err;
47     res.redirect('/');
48   });
49 });
50
```

8. Menampilkan form edit dan pembaruan data.

```
51 //update
52 app.get('/edit/:id', (req, res) => {
53   const query = 'SELECT * FROM users WHERE id = ?';
54   connection.query(query, [req.params.id], (err, result) => {
55     if (err) throw err;
56     res.render('edit', {users: result[0]});
57   });
58 });
59
node_modules > $ npm run dev
60
61 app.post('/update/:id', (req, res) => {
62   const { name, email, phone } = req.body;
63   const query = 'UPDATE users SET name=?, email=?, phone=? WHERE id=?';
64   connection.query(query, [name, email, phone, req.params.id], (err, result) => {
65     if (err) throw err;
66     res.redirect('/');
67   });
68 }
```

9. Menghapus data berdasarkan ID.

```
69 //hapus
70 app.get('/delete/:id', (req, res) => {
71   const query = 'DELETE FROM users WHERE id = ?';
72   connection.query(query, [req.params.id], (err, result) => {
73     if (err) throw err;
74     res.redirect('/');
75   });
76 });
77
```

10. Menjalankan server port localhost:3000

```
78 app.listen(3000, () => {
79   console.log("Server berjalan di port 3000, buka web melalui http://localhost:3000");
80 });
```

11. Membuat index.ejs

```
node_modules > views > index.ejs > html > body
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   <title>CRUD Node JS - MySQL</title>
7 </head>
8 <body>
9   <h1>Daftar User/Pengguna</h1>
10   <table border="1">
11     <tr>
12       <th>ID</th>
13       <th>Nama</th>
14       <th>Email</th>
15       <th>Telepon</th>
16       <th>Aksi</th>
17     </tr>
18     <% users.forEach(user => {%>
19       <tr>
20         <td><%=pengguna.id %></td>
21         <td><%=pengguna.name %></td>
22         <td><%=pengguna.email %></td>
23         <td><%=pengguna.phone %></td>
24         <td>
25           <a href="/edit/<%=pengguna.id %>">Edit</a>
26           <a href="/delete/<%=pengguna.id %>">Hapus</a>
27         </td>
28       </tr>
29     <% } %>
30   </table>
31
```

```

31
32 <h2>Tambah Pengguna Baru</h2>
33 <form action="/add" method="post">
34   <label for="Name">Nama:</label>
35   <input type="text" id="name" name="name" required><br>
36   <label for="Email">Email:</label>
37   <input type="email" name="email" id="email"><br>
38   <label for="phone">Telepon:</label>
39   <input type="text" name="phone" id="phone"><br>
40   <button type="submit">Tambah</button>
41 </form>
42 </body>
43 </html>

```

12. Membuat file edit.ejs



```

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   <title>Edit Data Pengguna</title>
7 </head>
8 <body>
9
10 <h2>Edit Data Pengguna Baru</h2>
11 <form action="/update/<%=user.id %>" method="post">
12   <label for="Name">Nama:</label>
13   <input type="text" id="name" name="name" required value="<%= user.name %>">
14   <label for="Email">Email:</label>
15   <input type="email" name="email" id="email" value="<%= user.email %>"><br>
16   <label for="phone">Telepon:</label>
17   <input type="text" name="phone" id="phone" value="<%= user.phone %>"><br>
18   <button type="submit">Simpan</button>
19 </form>
20 </body>
21 </html>
22

```

IV. KESIMPULAN

Dalam praktikum ini pemahaman mengenai CRUD sangat penting bagi developer. Dengan menguasai konsep ini, developer dapat merancang dan mengimplementasikan sistem yang mampu mengelola data secara efektif dan efisien. Meskipun terkesan sederhana, CRUD menjadi dasar bagi operasi yang lebih kompleks dalam developer aplikasi modern. Seiring berkembangnya teknologi, implementasi CRUD juga terus berkembang, namun prinsip dasarnya tetap sama dan terus menjadi bagian integral dari pengembangan aplikasi.

Link Github:

https://github.com/maharaninelita/TugasPRAK1_PemWeb.git