

Laporan Praktikum
Mata Kuliah Pemrograman Berorientasi Objek



Pertemuan 6
“CRUD: Create, Read, Update & Delete”

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

Disusun Oleh :
Dzakiya Fikri Murtianingsih
2300313

PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA
2024

I. PENDAHULUAN

Pada pertemuan ke-5 di Mata Kuliah Pemrograman WEB lalu, kami belajar mengenai CRUD: Create, Read, Update, dan Delete yang dilakukan di aplikasi Visual Code dengan alat atau aplikasi tambahan yang dibutuhkan, yaitu Node JS. CRUD biasa digunakan pada database untuk membuat, menampilkan, menambahkan, dan menghapus sebuah data. Sebelumnya kami telah mempelajari CRUD di PHPMyAdmin, namun kali ini kami mempraktikkannya di Visual Code dengan cara yang hampir sama.

Pertemuan ke-6 Mata Kuliah Pemrograman Berorientasi Objek ini melanjutkan materi minggu lalu pada Mata Kuliah Pemrograman WEB, yang dimana pada pertemuan ini kami ditugaskan untuk berkreasi membuat CRUD dengan tampilan web yang disempurnakan menggunakan CSS.

II. ALAT DAN BAHAN

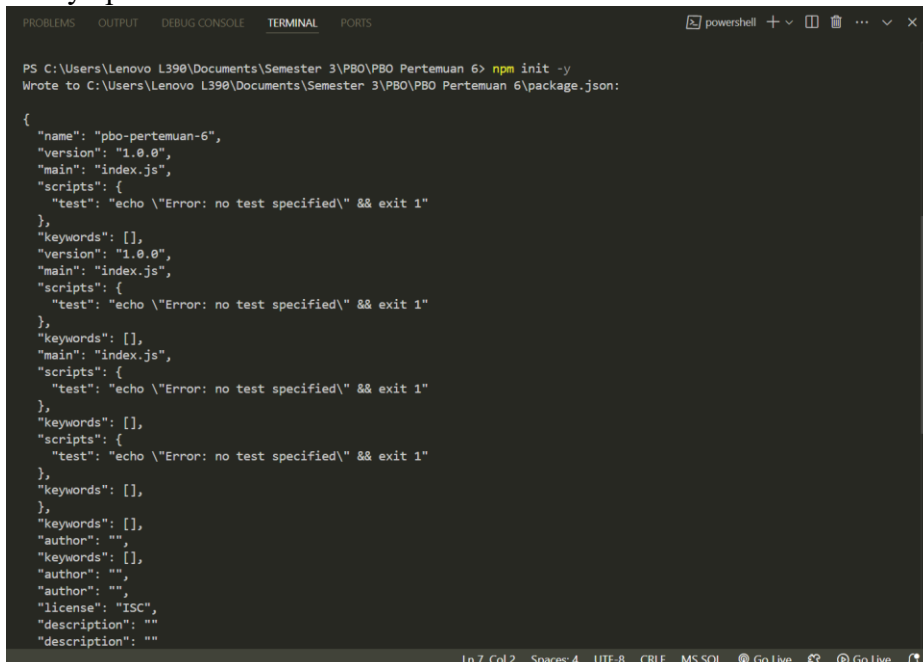
Alat yang digunakan pada praktikum ini adalah sebagai berikut:

1. Laptop/komputer.
2. Aplikasi Visual Code.
3. Aplikasi Node JS.

III. LANGKAH KERJA

Langkah-langkah yang dilakukan adalah sebagai berikut:

1. Buka aplikasi Visual Code, tambahkan folder baru, kemudian beralih ke aplikasi XAMPP untuk mengaktifkan MySQL. Setelah itu cek NPM dengan cara ketik “npm init -y” pada terminal.



```
PS C:\Users\Lenovo L390\Documents\Semester 3\PBO\PBO Pertemuan 6> npm init -y
Wrote to C:\Users\Lenovo L390\Documents\Semester 3\PBO\PBO Pertemuan 6\package.json:

{
  "name": "pbo-pertemuan-6",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "keywords": [],
  "author": "",
  "keywords": [],
  "author": "",
  "author": "",
  "license": "ISC",
  "description": ""
  "description": ""
}
```

2. Ketika sudah mengecek NPM-nya, kemudian install beberapa alat dengan mengetikkan tulisan “npm install express mysql2 body-parser ejs” seperti pada gambar.

```
PS C:\Users\Lenovo L390\Documents\Semester 3\PBO\PBO Pertemuan 6> npm install express mysql2 body-parser ejs
added 93 packages, and audited 94 packages in 15s

16 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
PS C:\Users\Lenovo L390\Documents\Semester 3\PBO\PBO Pertemuan 6> 
```

3. Buat database baru.

```
create-db-template.sql X create-table-template.sql
C: > Users > Lenovo L390 > .dbclient > storage > 1728733790575@@127.0.0.1@3306 >
1 -- Active: 1728733790575@@127.0.0.1@3306 MySQL XAMPP
  Run | New Tab
2 CREATE DATABASE PBO_Pertemuan6
3 DEFAULT CHARACTER SET = 'utf8mb4'; 26ms AffectedRows: 1
```

4. Buat tabel baru.

```
C: > Users > Lenovo L390 > .dbclient > storage > 1728733790575@@127.0.0.1@3306@pbo_pertemuan6 >
1 -- Active: 1728733790575@@127.0.0.1@3306@pbo_pertemuan6 MySQL XAMPP
  Run | Reset
2 CREATE Table anggota(
3   id INT AUTO_INCREMENT PRIMARY KEY,
4   name VARCHAR(30),
5   instagram VARCHAR(20),
6   grup VARCHAR(15)
7 ) 157ms
```

5. Isi data pada tabel yang telah dibuat sebelumnya.

Properties DATA Diagram Monitor Log

SELECT * FROM anggota LIMIT 100

Search results

	* id int(11)	name varchar(30)	instagram varchar(20)	grup varchar(15)
	Filter	Filter	Filter	Filter grup
>	1	Mark Lee	onyourm_ark	NCT Dream & 12
>	2	Na Jaemin	na.jaemin0813	NCT Dream
>	3	Lee Jeno	leejen_o_423	NCT Dream
>	4	Lee Taeyong	taeoxo_nct	NCT 127

Cost: 7ms < 1 > Total 4

6. Kemudian beralih ke explorer untuk membuat file baru, yaitu: app.js, index.ejs, dan edit.ejs.

IV. PENJELASAN KODE PROGRAM

Kode Program:

// memanggil library yang sudah diinstall sebelumnya di terminal.

```
const express = require('express');
```

```
const mysql = require('mysql2');
```

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

```

// memanggil atau menjalankan fungsi dari library express.
// menggunakan bodyparser untuk parsing informasi yang ada dalam url
const app = express();

app.use(bodyParser.urlencoded({extended : false})); //urlencoded itu untuk mengambil
data url seperti www, dsb.

app.use(bodyParser.json()); //json digunakan untuk pertukaran data.


// membuat koneksi MySQL, sesuaikan isinya seperti XAMPP
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'pbo_pertemuan6'
});

// untuk melihat datanya dapat berjalan atau tidak
connection.connect((err) =>{
  if(err){
    console.error("Terjadi kesalahan dalam koneksi ke MySQL:", err.stack);
    return;
  }
  console.log("Koneksi MySQL BERHASIL dengan id" + connection.threadId)
});

// digunakan untuk routing, yaitu memberikan informasi kemana rute dari app.js akan
membuka filenya.
// macam-macam routing: create, read, update, dan delete.
app.set('view engine', 'ejs');

// get digunakan untuk mengambil data
// READ
app.get('/', (req, res) =>{
  const query = 'SELECT * FROM anggota';
  connection.query(query, (err, results) =>{

```

```
    res.render('index', {anggota: results});
  });
}); //untuk menampilkan data pada halaman index.
```

// CREATE atau INPUT

```
app.post('/add', (req, res) => {
  const { name, instagram, grup } = req.body;
  const query = 'INSERT INTO anggota (name, instagram, grup) VALUES (?, ?, ?)';
  connection.query(query, [name, instagram, grup], (err, results) => {
    if(err) throw err;
    res.redirect('/');
  });
});
```

// UPDATE

// untuk akses halaman

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

// untuk update data

```
app.post('/update/:id', (req, res) => {
  const { name, instagram, grup } = req.body;
  const query = 'UPDATE anggota SET name = ?, instagram = ?, grup = ? WHERE id = ?';
  connection.query(query, [name, instagram, grup, req.params.id], (err, results) => {
    if(err) throw err;
    res.redirect('/');
  });
});
```

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

app.listen(127,()=>{
  console.log("Server berjalan di port 3000, buka web melalui http://localhost:127")
});
```

Penjelasan:

- Pertama memanggil library yang sudah diinstall sebelumnya di terminal, yaitu express, mysql2, dan body-parser.
- Menjalankan fungsi dari library express dengan cara “const app = express();”, kemudian diikuti dengan “app.use(bodyParser.urlencoded({extended : false}));” dan “app.use(bodyParser.json());”.

```
const app = express();
app.use(bodyParser.urlencoded({extended : false})); //urlencoded digunakan
untuk mengambil data url seperti www, dsb.
app.use(bodyParser.json()); //json digunakan untuk pertukaran data.
```

- Membuat koneksi MySQL yang nantinya digunakan untuk mengakses database.

```
const connection = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'pbo_pertemuan6'
});
```

- Membuat sintaks untuk melihat atau memberikan pesan apabila terjadi error ketika membuka database.

```
connection.connect((err) =>{
  if(err){
    console.error("Terjadi kesalahan dalam koneksi ke MySQL:",
err.stack);
    return;
  }
  console.log("Koneksi MySQL BERHASIL dengan id" + connection.threadId)
});
```

- Membuat sintaks untuk routing.

```
// digunakan untuk routing, yaitu memberikan informasi kemana rute dari
app.js akan membuka filenya.
```

```
// macam-macam routing: create, read, update, dan delete.
app.set('view engine', 'ejs');
```

- Membuat sintaks untuk menampilkan routing tersebut:

a. Read

```
app.get('/', (req, res) =>{
  const query = 'SELECT * FROM anggota';
  connection.query(query, (err, results) =>{
    res.render('index', {anggota: results});
  });
});
```

b. Create

```
app.post('/add', (req, res) =>{
  const { name, instagram, grup } = req.body;
  const query = 'INSERT INTO anggota (name, instagram, grup) VALUES
  (?, ?, ?)';
  connection.query(query, [name, instagram, grup], (err, results) =>{
    if(err) throw err;
    res.redirect('/');
  });
});
```

c. Update

```
// untuk akses halaman
app.get('/edit/:id', (req, res) => {
  const query = 'SELECT * FROM anggota WHERE id = ?';
  connection.query(query, [req.params.id], (err, result) =>{
    if(err) throw err;
    res.render('edit', {anggota: result[0]});
  });
});
// untuk update data
app.post('/update/:id', (req, res) =>{
  const { name, instagram, grup } = req.body;
  const query = 'UPDATE anggota SET name = ?, instagram = ?, grup = ?
  WHERE id = ?';
  connection.query(query, [name, instagram, grup, req.params.id],
  (err, results) =>{
    if(err) throw err;
    res.redirect('/');
  });
});
```

Get dan post memiliki perbedaan, get digunakan untuk menampilkan data, sedangkan post digunakan untuk memposting data hasil dari edit.

d. Delete

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

```
});  
});
```

- Membuat link untuk menampilkan di halaman web.

```
app.listen(127,()=>{  
  console.log("Server berjalan di port 3000, buka web melalui  
http://localhost:127")  
});
```

Hasil:

a. Menambahkan

The screenshot shows a web browser at localhost:127 displaying a page titled "Daftar Anggota Boy Grup SM". It contains a table with 6 members and a form to add a new member. The second screenshot shows the same page after adding a new member, "Ning Ning", to the table.

ID	NAMA	INSTAGRAM	GRUP	AKSI
1	Lee Minhyung	onyourm_ark	NCT Dream & 127	Edit Hapus
2	Na Jaemin	na.jaemin0813	NCT Dream	Edit Hapus
3	Lee Jeno	leejen_o_423	NCT Dream	Edit Hapus
4	Lee Taeyong	taeoxo_nct	NCT 127	Edit Hapus
5	Winwin	wwiinn_7	WayV	Edit Hapus
6	Kim Doyoung	do0_nct	NCT 127	Edit Hapus

Tambah Anggota Baru

Nama:

Instagram:

Grup:

ID	NAMA	INSTAGRAM	GRUP	AKSI
1	Lee Minhyung	onyourm_ark	NCT Dream & 127	Edit Hapus
2	Na Jaemin	na.jaemin0813	NCT Dream	Edit Hapus
3	Lee Jeno	leejen_o_423	NCT Dream	Edit Hapus
4	Lee Taeyong	taeoxo_nct	NCT 127	Edit Hapus
5	Winwin	wwiinn_7	WayV	Edit Hapus
6	Kim Doyoung	do0_nct	NCT 127	Edit Hapus
7	Ning Ning	ningning	Aespa	Edit Hapus

Tambah Anggota Baru

Nama:

Instagram:

Grup:

b. Mengedit

The screenshot shows a web browser at localhost:127/edit/7. The page title is "Edit Anggota". It contains a form with the following fields:

- Nama: Karina
- Instagram: karina
- Grup: Aespa

A dropdown menu is open for the "Grup" field, showing a list of groups: Aespa, Aespa EN, Aesap, Aesha, and Aesha EN. The "Simpan" button is visible.

Below the form, there is a table titled "Daftar Anggota Boy Grup SM".

ID	NAMA	INSTAGRAM	GRUP	AKSI
1	Lee Minhyung	onyourm_ark	NCT Dream & 127	Edit Hapus
2	Na Jaemin	na.jaemin0813	NCT Dream	Edit Hapus
3	Lee Jeno	leejen_o_423	NCT Dream	Edit Hapus
4	Lee Taeyong	taeoxo_nct	NCT 127	Edit Hapus
5	Winwin	wwiinn_7	WayV	Edit Hapus
6	Kim Doyoung	do0_nct	NCT 127	Edit Hapus
7	Karina	karina	Aespa	Edit Hapus

Below the table, there is a form titled "Tambah Anggota Baru".

Nama:

Instagram:

Grup:

c. Menghapus (langsung klik “hapus”)

The screenshot shows a web browser at localhost:127. The page title is "Daftar Anggota Boy Grup SM". It contains a table with the following data:

ID	NAMA	INSTAGRAM	GRUP	AKSI
1	Lee Minhyung	onyourm_ark	NCT Dream & 127	Edit Hapus
2	Na Jaemin	na.jaemin0813	NCT Dream	Edit Hapus
3	Lee Jeno	leejen_o_423	NCT Dream	Edit Hapus
4	Lee Taeyong	taeoxo_nct	NCT 127	Edit Hapus
5	Winwin	wwiinn_7	WayV	Edit Hapus

Below the table, there is a form titled "Tambah Anggota Baru".

Nama:

Instagram:

Grup: