

Laporan Praktikum
Mata Kuliah Pemrograman Web



Pertemuan 5
"Praktikum 4"

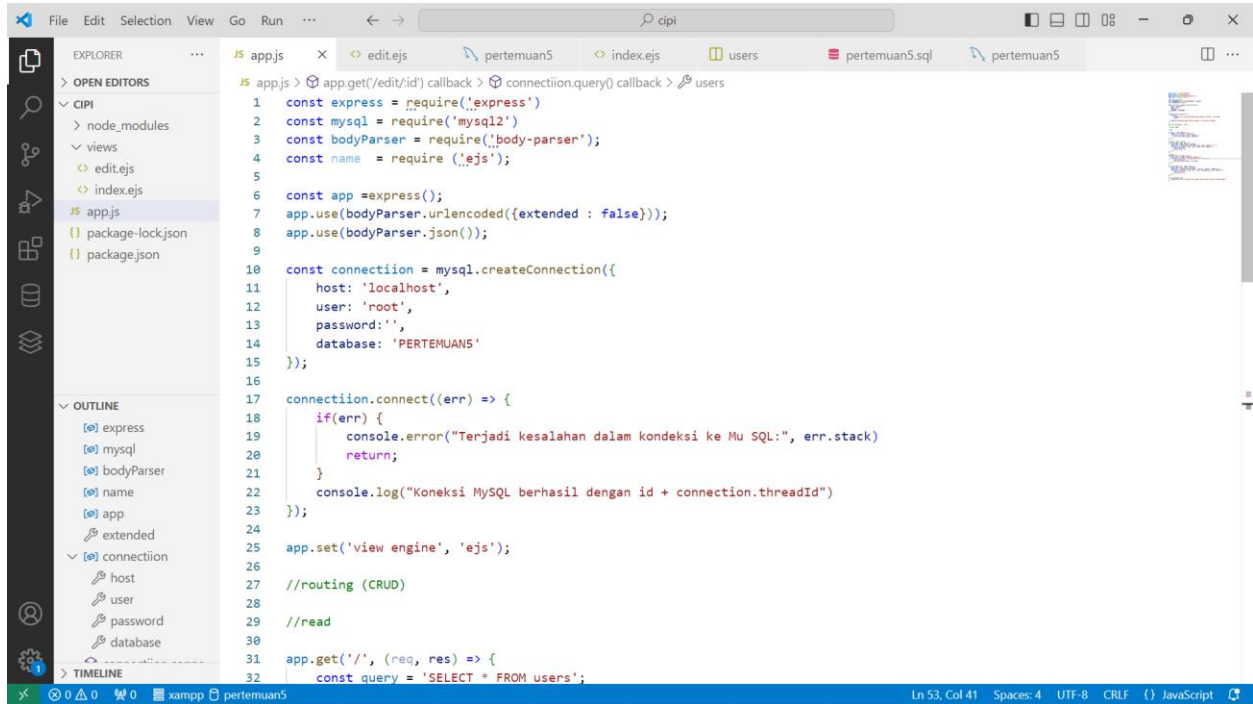
Dosen Pengampu :
Wildan Aprizal Arifin, S.Pd., M.Kom.

Disusun Oleh :
Silvia Isti Lestary
2311883

PROGRAM STUDI SISTEM INFORMASI KELAUTAN
UNIVERSITAS PENDIDIKAN INDONESIA
2024

1. Koneksi ke Database MySQL

Membuat library MySQL2 dan membuat koneksi dengan `mysql.createConnection`.



The screenshot shows a VS Code editor with a file explorer on the left and a code editor in the center. The file explorer shows a project structure with files like `package-lock.json` and `package.json`. The code editor displays the following JavaScript code:

```
1 const express = require('express')
2 const mysql = require('mysql2')
3 const bodyParser = require('body-parser');
4 const name = require('ejs');
5
6 const app = express();
7 app.use(bodyParser.urlencoded({extended: false}));
8 app.use(bodyParser.json());
9
10 const connection = mysql.createConnection({
11   host: 'localhost',
12   user: 'root',
13   password: '',
14   database: 'PERTEMUAN5'
15 });
16
17 connection.connect((err) => {
18   if(err) {
19     console.error("Terjadi kesalahan dalam koneksi ke MySQL:", err.stack)
20     return;
21   }
22   console.log("Koneksi MySQL berhasil dengan id + connection.threadId")
23 });
24
25 app.set('view engine', 'ejs');
26
27 //routing (CRUD)
28
29 //read
30
31 app.get('/', (req, res) => {
32   const query = 'SELECT * FROM users';
```

2. Routing

Yaitu mengambil daftar pengguna dari database, menggunakan query MySQL `SELECT * FROM users`, ditampilkan menggunakan template `ejs`.

```
30
31 app.get('/', (req, res) => {
32   const query = 'SELECT * FROM users';
33   connection.query(query, (err, results) => {
34     res.render('index', {users: results});
35   });
36 });
37
```

3. Create

Setelah data di submit, data akan dikirim melalui post, menggunakan query MySQL `INSERT`

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

4. Update

Sistem mengambil data berdasarkan id, menampilkan form edit, dan mengupdate data ke database.

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

5. Delete

Pengguna dapat menghapus database dengan query DELETE FROM users WHERE id = ?

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

6. Template Engine

Menggunakan EJS untuk menghasilkan data dinamis di halaman HTML.

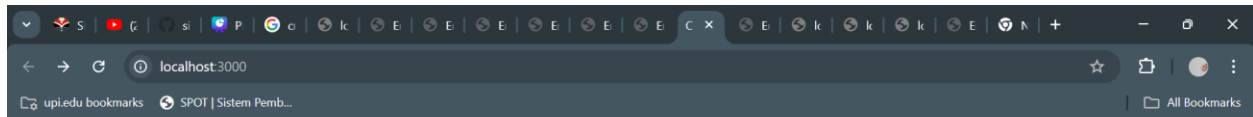
```

<% users.forEach(Pengguna => {%>
<tr>
<td><%= Pengguna.id %></td>
<td><%= Pengguna.name %></td>
<td><%= Pengguna.email %></td>
<td><%= Pengguna.phone %></td>
<td>
<a href="/edit/<%= Pengguna.id%>">Edit</a>
<a href="/delete/<%= Pengguna.id%>">Hapus</a>
</td>
</tr>
<% }) %>

```

HASIL

Tampilan dari aplikasi CRUD di browser.



Daftar user/Pengguna

ID	Nama	Email	Telepon	Aksi
1	Silvia	silviaa@upi.edu	123456789	Edit Hapus
2	Bintang	bintangg@upi.edu	123456789	Edit Hapus
3	Faiq	faiqq@uppi.edu	123456789	Edit Hapus
4	Sabrina	sabrinaa@upi.edu	23456789	Edit Hapus

Tambah pengguna baru

Nama:

Email:

Telepon:

