



D3 SISTEM INFORMASI

PEMROGRAMAN WEB LANJUT

2025



LEMBAR PENGESAHAN

Modul Pembelajaran Pemrograman Web Lanjut

Oleh:

Muhammad Panji Muslim, S.Pd., M.Kom

NIP. 199308312022031007

Modul ini disusun sebagai pedoman dan acuan dalam pelaksanaan
Praktikum Pemrograman Web Semester VI
Program Studi D3 Sistem Informasi, Fakultas Ilmu Komputer
Universitas Pembangunan Nasional “Veteran” Jakarta

Disahkan pada tanggal 10 Februari 2025

Kepala Program Studi D3 Sistem Informasi

NIP.

KATA PENGANTAR

Alhamdulillah rabbil'alamin. Puji dan syukur senantiasa penyusun panjatkan kehadirat Allah SWT yang telah melimpahkan rahmat dan karunia-Nya sehingga penyusun dapat menyelesaikan Modul Praktikum Pemrograman Web sebagai pusat kegiatan belajar mahasiswa. Modul ini juga dilengkapi dengan Langkah-langkah dari Instalasi XAMPP sampai implementasi pemrograman HTML, CSS, Javascript, dan PHP.

Penyusun menyadari masih banyak kekurangan dalam penyusunan modul ini. Oleh karena itu, kami sangat mengharap kritik dan saran demi perbaikan dan optimalisasi modul ini.

Penyusun mengucapkan terima kasih kepada berbagai pihak yang telah membantu proses penyelesaian modul ini. Semoga modul ini dapat bermanfaat bagi kita semua, khususnya para mahasiswa Pemrograman Web Sistem Informasi.

Jakarta, 10 Februari 2025

Penyusun

TATA TERTIB PEMBELAJARAN

1. Mahasiswa yang diperkenankan menggunakan laboratorium dan melakukan praktikum adalah mahasiswa yang terdaftar secara akademik (praktikan).
2. Praktikan wajib hadir 15 menit sebelum praktikum dimulai, keterlambatan lebih dari 5 menit sejak praktikum dimulai, praktikan dianggap tidak hadir.
3. Jika berhalangan hadir, praktikan harus dapat memberikan keterangan tertulis dan resmi terkait dengan alasan ketidakhadirannya.
4. Jika berhalangan hadir dan hendak mengganti praktikum pada hari yang lain, praktikan wajib meminta rekomendasi tertulis terlebih dahulu dari koordinator pembimbing praktikum.
5. Praktikan wajib membawa lembar kerja praktikum dan memakai masker.
6. Praktikan mengisi daftar absensi dengan menunjukkan segala sesuatu yang wajib dibawa.
7. Praktikan tidak diperbolehkan makan, minum, atau merokok di dalam laboratorium selama praktikum berlangsung.
8. Praktikan tidak diperbolehkan bersenda gurau yang mengakibatkan terganggunya kelancaran praktikum.
9. Praktikan bertanggung jawab atas peralatan yang dipinjamnya, kebersihan meja masing-masing, serta lantai disekitarnya.
10. Setelah menggunakan peralatan laboratorium, praktikan wajib meletakkan kembali pada tempatnya semula.
11. Jika akan meninggalkan ruang laboratorium, praktikan wajib meminta izin kepada dosen atau asisten jaga.



MATERI XI

Menghubungkan Frontend dan Backend

Tujuan Pembelajaran

Pada pertemuan ini, Mahasiswa dapat memahami dan mempraktikkan integrasi antara frontend (React.js) dan backend (Node.js + Express) yang terhubung dengan database MySQL melalui studi kasus manajemen produk dan transaksi pembayaran.

1. Konsep Dasar (*Frontend dan Backend*)

Frontend dan backend. Frontend, dalam konteks ini dibangun menggunakan React.js, berfungsi untuk menampilkan antarmuka pengguna, menangkap input yang dimasukkan oleh pengguna, dan mengirimkan data tersebut ke backend melalui permintaan HTTP. Untuk komunikasi ini, digunakan library seperti axios yang mempermudah pengiriman dan penerimaan data antar sisi klien dan server.

Sementara itu, backend menggunakan Node.js dengan framework Express untuk menangani permintaan dari frontend. Di sisi backend inilah semua logika bisnis dijalankan, seperti memproses data yang masuk, melakukan validasi, serta berinteraksi dengan sistem penyimpanan data yaitu database MySQL. Untuk mengakses MySQL, backend biasanya menggunakan library seperti mysql2 yang memungkinkan eksekusi perintah SQL seperti INSERT, SELECT, dan DELETE dengan mudah dan efisien.

Dalam implementasi yang dipelajari pada pertemuan ini, terdapat dua entitas utama yang dikelola: produk dan pembayaran. Ketika pengguna menambahkan produk melalui form yang tersedia di antarmuka React, data tersebut dikirim ke backend dan disimpan ke dalam tabel produk pada database MySQL. Produk-produk yang telah disimpan dapat ditampilkan kembali di antarmuka dengan melakukan permintaan GET dari frontend ke backend, lalu backend mengambil data dari database dan mengirimkannya kembali ke frontend.

Proses lainnya adalah transaksi pembayaran. Pengguna dapat memilih salah satu produk, mengisi data nama pembeli dan jumlah produk yang dibeli. Informasi ini kemudian dikirim dari React ke backend. Backend akan memproses data tersebut, menghitung total harga berdasarkan jumlah dan harga satuan produk, lalu menyimpan data transaksi ini ke dalam tabel pembayaran di database MySQL.

Dengan pemahaman menyeluruh terhadap hubungan antara frontend, backend, dan database, mahasiswa diharapkan mampu membangun aplikasi sederhana yang menggabungkan antarmuka pengguna dengan proses penyimpanan data dan pengolahan transaksi secara terintegrasi.

Untuk Lebih Jelasnya marilah kita coba

Langkah Langkah Implementasi “Advance Asynchronous”

Langkah 1. Setup lingkungan pengembangan

Sebelum membuat project pastikan Node.js sudah terinstall di komputer. Jika belum, download dari nodejs.org, lalu apabila telah terinstall di komputer cek instalasi dengan menjalankan perintah di terminal:

```
muhammadpanji@Muhammads-MacBook-Pro-4 ~ % node -v
v22.2.0
muhammadpanji@Muhammads-MacBook-Pro-4 ~ % npm -v
10.7.0
```

Gambar 1. Cek Instalasi Node.js

Langkah 2. Struktur Proyek

Langkah selanjutnya menyamai Struktur Direktori sebagai berikut:

Backend

```
p11_be/
├── db.js
├── server.js
├── routes/
│   ├── produk.js
│   └── pembayaran.js
```

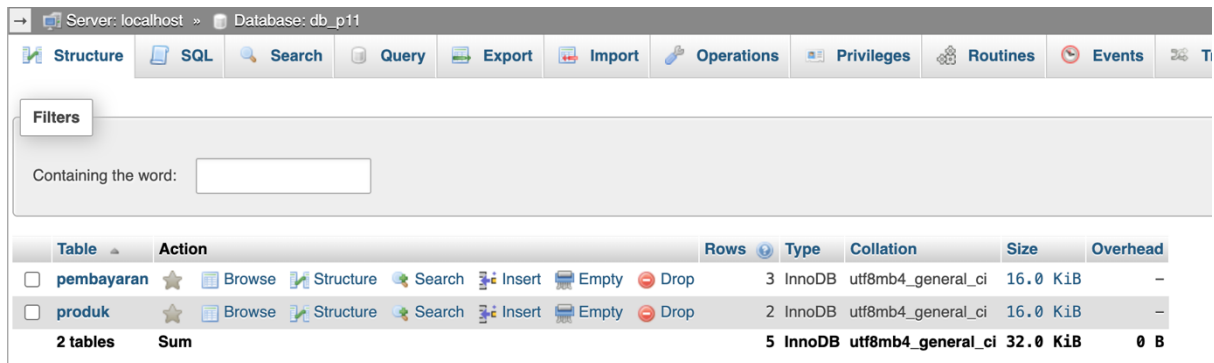
```
p11_fe/
├── node_modules/
├── public/
│   └── (index.html dll.) # Entry point HTML React
├── src/
│   ├── App.css
│   ├── App.js
│   ├── App.test.js
│   ├── index.css
│   ├── index.js
│   ├── logo.svg
│   ├── ProdukList.js # Komponen Produk berisi produk dan pembayaran
│   ├── reportWebVitals.js
│   └── setupTests.js
```

Langkah 3. Inisialisasi Project

Buat folder baru untuk pertemuan ini dengan nama frontend dan backen **p11_be** (**mkdir p11_be**) dan **p11_fe** inisialisasi Node.js (**Untuk frontend buat menggunakan React npx create-react-app p11_fe**).

Langkah 4. Buat Database dan Table pada MYSQL

Untuk kebutuhan manipulasi data buatlah db dengan nama **db_p11** dan table seperti berikut:



The screenshot shows the MySQL Workbench interface for a database named 'db_p11' on a local server. The 'Structure' tab is active, displaying a list of tables. Below the table list, there is a 'Filters' section with a search box labeled 'Containing the word:'. The table list includes columns for 'Table', 'Action', 'Rows', 'Type', 'Collation', 'Size', and 'Overhead'. Two tables are listed: 'pembayaran' and 'produk'. The 'pembayaran' table has 3 rows, and the 'produk' table has 2 rows. The total for both tables is 5 rows.

Table	Action	Rows	Type	Collation	Size	Overhead
<input type="checkbox"/> pembayaran	★ Browse Structure Search Insert Empty Drop	3	InnoDB	utf8mb4_general_ci	16.0 KiB	-
<input type="checkbox"/> produk	★ Browse Structure Search Insert Empty Drop	2	InnoDB	utf8mb4_general_ci	16.0 KiB	-
2 tables	Sum	5	InnoDB	utf8mb4_general_ci	32.0 KiB	0 B

Gambar 2. Database

Dengan struktur table sebagai berikut

```
CREATE DATABASE db_p11;
```

```
USE db_p11;
```

```
CREATE TABLE produk (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    nama VARCHAR(100),  
    harga INT  
);
```

```
CREATE TABLE pembayaran (  
    id INT AUTO_INCREMENT PRIMARY KEY,  
    nama_pembeli VARCHAR(100),  
    nama_produk VARCHAR(100),  
    jumlah INT,  
    total_harga INT,  
    tanggal DATETIME DEFAULT CURRENT_TIMESTAMP  
);
```

Langkah 4. Buat konfigurasi database (backend)

Untuk backend buatlah file db.js sebagai konfigurasi database dengan code sebagai berikut:

```
const mysql = require('mysql2');
const db = mysql.createConnection({
  host: 'localhost',
  user: 'root',
  password: '',
  database: 'db_p11' // nama database
});

db.connect(err => {
  if (err) throw err;
  console.log('MySQL Connected');
});

module.exports = db;
```

Langkah 5. Buat REST API (routes/produk.js)

Untuk backend buatlah file produk.js sebagai REST API produk dengan code sebagai berikut:

```
const express = require('express');
const router = express.Router();
const db = require('../db');

// GET: ambil semua produk
router.get('/', (req, res) => {
  db.query('SELECT * FROM produk', (err, results) => {
    if (err) return res.status(500).json(err);
    res.json(results);
  });
});

// POST: tambah produk baru
router.post('/', (req, res) => {
  const { nama, harga } = req.body;
  db.query('INSERT INTO produk (nama, harga) VALUES (?, ?)', [nama, harga], (err, result) => {
    if (err) return res.status(500).json(err);
    res.json({ id: result.insertId, nama, harga });
  });
});
```



```

});

// DELETE: hapus produk
router.delete('/:id', (req, res) => {
  db.query('DELETE FROM produk WHERE id = ?', [req.params.id],
  (err) => {
    if (err) return res.status(500).json(err);
    res.json({ message: 'Produk dihapus' });
  });
});

module.exports = router;

```

Langkah 6. Buat REST API (routes/pembayaran.js)

Untuk backend buatlah file pembayaran.js sebagai REST API produk dengan code sebagai berikut:

```

const express = require('express');
const router = express.Router();
const db = require('../db');

// GET: semua pembayaran
router.get('/', (req, res) => {
  db.query('SELECT * FROM pembayaran ORDER BY tanggal DESC',
  (err, results) => {
    if (err) return res.status(500).json(err);
    res.json(results);
  });
});

// POST: tambah pembayaran
router.post('/', (req, res) => {
  const { nama_pembeli, nama_produk, jumlah, total_harga } =
  req.body;
  db.query(
    'INSERT INTO pembayaran (nama_pembeli, nama_produk,
  jumlah, total_harga) VALUES (?, ?, ?, ?)',
    [nama_pembeli, nama_produk, jumlah, total_harga],
    (err, result) => {
      if (err) return res.status(500).json(err);
      res.json({ message: 'Pembayaran berhasil dicatat' });
    }
  );
});

module.exports = router;

```

Langkah 7. Buat Server Utama (server.js)

Untuk backend buatlah server.js sebagai server utama dengan code sebagai berikut:

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

const app = express();
app.use(cors());
app.use(bodyParser.json());

const produkRoutes = require('./routes/produk');
app.use('/api/produk', produkRoutes);

const pembayaranRoutes = require('./routes/pembayaran');
app.use('/api/bayar', pembayaranRoutes);

app.listen(5000, 'localhost', () => {
  console.log('Server berjalan di http://localhost:5000');
});
```

Langkah 8. Buat File pada Folder Frontend

Buat File p11_fe/src/ProdukList.js terlebih dahulu dengan source code berikut:

```
import { useEffect, useState } from 'react';
import axios from 'axios';

function ProdukList() {
  const [products, setProducts] = useState([]);
  const [form, setForm] = useState({ nama: '', harga: '' });
  const [pembeli, setPembeli] = useState('');
  const [jumlah, setJumlah] = useState(1);
  const [produkDipilih, setProdukDipilih] = useState(null);

  const fetchProducts = async () => {
    const res = await
    axios.get('http://localhost:5000/api/produk');
    setProducts(res.data);
  };

  const addProduct = async () => {
    await axios.post('http://localhost:5000/api/produk',
form);
    fetchProducts();
    setForm({ nama: '', harga: '' });
  };

  const deleteProduct = async (id) => {
```

```

    await
    axios.delete(`http://localhost:5000/api/produk/${id}`);
    fetchProducts();
  };

  const handleBayar = async () => {
    if (!produkDipilih || !pembeli || !jumlah) return
    alert("Lengkapi semua data!");

    const total_harga = produkDipilih.harga * jumlah;

    await axios.post('http://localhost:5000/api/bayar', {
      nama_pembeli: pembeli,
      nama_produk: produkDipilih.nama,
      jumlah: jumlah,
      total_harga: total_harga,
    });

    alert("Pembayaran berhasil!");
    setPembeli('');
    setJumlah(1);
    setProdukDipilih(null);
  };

  useEffect(() => {
    fetchProducts();
  }, []);

  return (
    <div>
      <h2>Daftar Produk</h2>

      <input
        type="text"
        placeholder="Nama Produk"
        value={form.nama}
        onChange={(e) => setForm({ ...form, nama:
e.target.value })}
      />
      <input
        type="number"
        placeholder="Harga"
        value={form.harga}
        onChange={(e) => setForm({ ...form, harga:
e.target.value })}
      />
      <button onClick={addProduct}>Tambah</button>
    </div>
  );

```

```

<ul>
  {products.map((p) => (
    <li key={p.id}>
      {p.nama} - Rp{p.harga}
      <button onClick={() =>
deleteProduct(p.id)}>Hapus</button>
      <button onClick={() =>
setProdukDipilih(p)}>Bayar</button>
    </li>
  ) )}
</ul>

{produkDipilih && (
  <div style={{ marginTop: '20px' }}>
    <h3>Pembayaran untuk: {produkDipilih.nama}</h3>
    <input
      type="text"
      placeholder="Nama Pembeli"
      value={pembeli}
      onChange={(e) => setPembeli(e.target.value)}
    />
    <input
      type="number"
      placeholder="Jumlah"
      value={jumlah}
      onChange={(e) =>
setJumlah(parseInt(e.target.value))}
      min="1"
    />
    <p>Total Harga: Rp{produkDipilih.harga * jumlah}</p>
    <button onClick={handleBayar}>Bayar
Sekarang</button>
  </div>
  )}
</div>
  );
}

export default ProdukList;

```

Lalu update File `p11_fe/src/App.js` terlebih dahulu dengan source code berikut:

```
import ProdukList from './ProdukList';

function App() {
  return (
    <div className="App">
      <h1>Manajemen Produk</h1>
      <ProdukList />
    </div>
  );
}

export default App;
```

Langkah 9. Jalankan Server

Berikut command untuk menjalankan Node.Js dan Express (Backend):

```
muhammadpanji@Muhammads-MacBook-Pro-4 p11_be % node server.js
Server berjalan di http://localhost:5000
MySQL Connected
█
```

Gambar 3. Menjalankan server backend

Berikut command untuk menjalankan React:

```
muhammadpanji@Muhammads-MacBook-Pro-4 frontend % npm start

Compiled successfully!

You can now view p11_fe in the browser.

  Local:                http://localhost:3000
  On Your Network:      http://172.20.10.11:3000

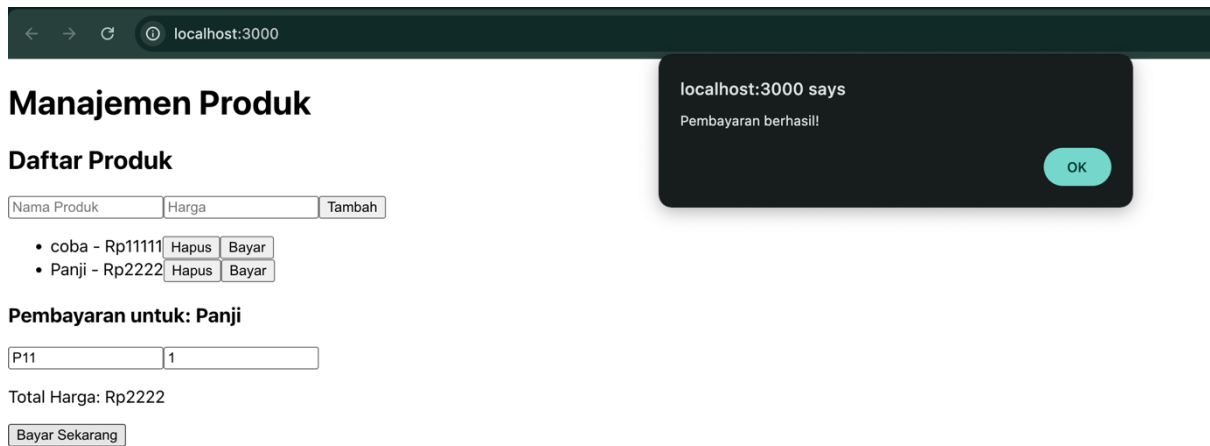
Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

Gambar 4. Menjalankan server frontend

Langkah 10. Uji coba

Jika keduanya berhasil react akan otomatis membuka browser lalu hit url seperti berikut:



Gambar 5.Ujicoba

Selamat Eksplorasi