

Nama: Yosan Pratiwi

NIM: 10231091

UTS PROWEB

# Aplikasi CRUD dengan React, Express, dan PostgreSQL

---

Aplikasi ini merupakan proyek latihan untuk mengimplementasikan fitur CRUD menggunakan **React (Vite)** di frontend, **Express.js** di backend, dan **PostgreSQL** sebagai database.

---

## ❖ Fitur Aplikasi

- Menampilkan daftar produk dengan tampilan responsif
  - Menambahkan produk baru dengan validasi input
  - Mengedit produk yang sudah ada dengan modal form
  - Menghapus produk dengan konfirmasi sebelum eksekusi
  - Menerapkan notifikasi sukses/gagal pada operasi CRUD
  - Menghubungkan frontend dan backend menggunakan API
  - Menampilkan data dalam tabel dengan fitur pencarian dan pagination
  - Menggunakan state management dengan React Hooks (`useState, useEffect`)
- 

## 🚀 Cara Install dan Menjalankan Aplikasi

### 1. Clone Repository

```
git clone https://github.com/yosanpratiwi/ecommerce-kecil.git  
cd repository-name
```

### 2. Install Dependensi

#### Backend

```
cd backend  
npm install
```

#### Frontend

```
cd frontend  
npm install
```

### 3. Konfigurasi Database

1. Pastikan PostgreSQL sudah terinstall.
2. Buat database dengan nama `ecommerce_kecil`.
3. Sesuaikan konfigurasi di file `db.js`.
4. Jalankan migrasi database:

```
cd backend  
node migrate.js
```

### 4. Menjalankan Aplikasi

#### Menjalankan Backend

```
cd backend  
node index.js
```

Backend akan berjalan di `http://localhost:3001`.

#### Menjalankan Frontend

```
cd frontend  
npm run dev
```

Frontend akan berjalan di `http://localhost:5173`.

## ❖ Daftar Endpoint API

### 1. Endpoint Produk

Method	Endpoint	Deskripsi
GET	/api/produk	Mendapatkan semua data produk
GET	/api/produk/:id	Mendapatkan satu produk berdasarkan ID
POST	/api/produk	Menambahkan produk baru
PUT	/api/produk/:id	Mengupdate produk berdasarkan ID

Method	Endpoint	Deskripsi
DELETE	/api/produk/:id	Menghapus produk berdasarkan ID

## 2. Contoh Payload untuk POST dan PUT

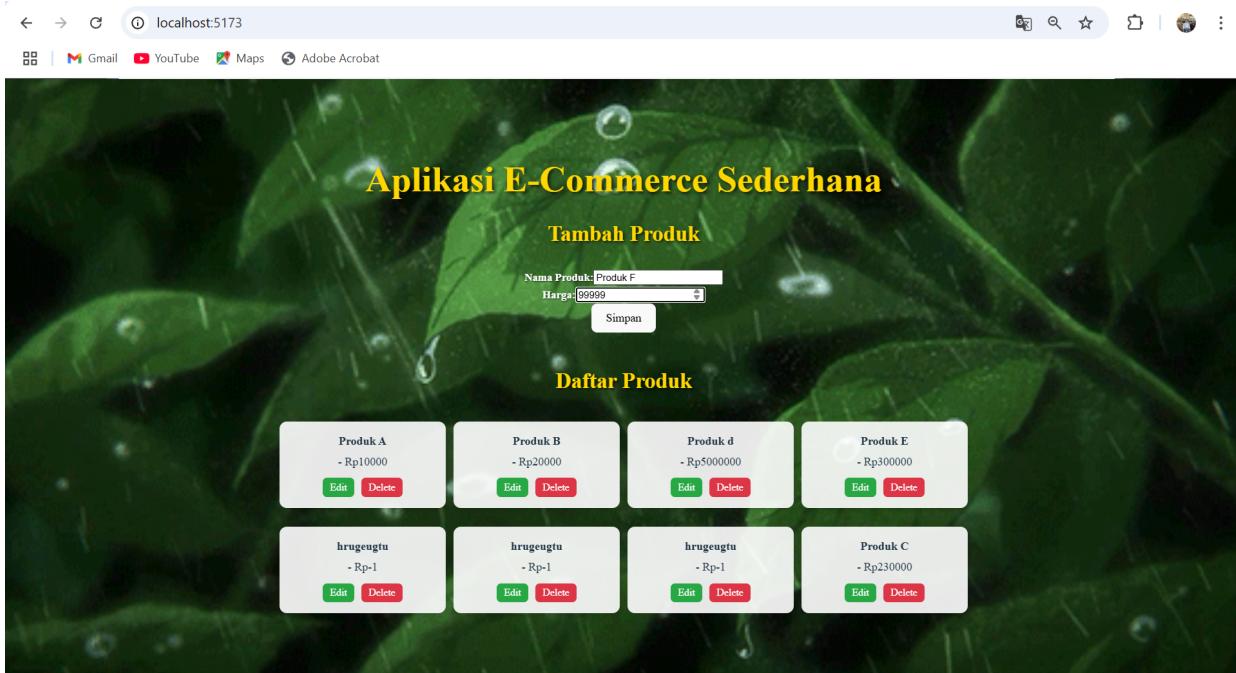
```
{
  "nama": "Produk A",
  "harga": 10000
}
```

## ⌚ Teknologi yang Digunakan

- **Frontend:** React (Vite), Tailwind CSS, Axios, React Hooks
- **Backend:** Express.js, Node.js, CORS
- **Database:** PostgreSQL, pgAdmin
- **Tools:** Postman, GitHub, Railway, Vercel
- **State Management:** useState, useEffect, Context API
- **Deployment:** GitHub Actions, Vercel, Railway

## 📷 Screenshot Aplikasi

### 1. Tampilan Tambah Produk dengan Form Validasi



## 2. Tampilan Read Data dalam Tabel dengan Pagination

The screenshot shows a web browser window with the URL `localhost:5173`. The page title is "Aplikasi E-Commerce Sederhana". The main content area has a green leafy background. At the top, there is a form titled "Tambah Produk" with fields for "Nama Produk" and "Harga", and a "Simpan" button. Below this is a section titled "Daftar Produk" containing a table with six rows of products. Each row contains a product name, its price, and two buttons: "Edit" and "Delete". The products listed are:

Produk	Harga
Produk A	- Rp10000
Produk B	- Rp20000
Produk d	- Rp5000000
Produk E	- Rp300000
hrugeugtu	- Rp-1
hrugeugtu	- Rp-1
hrugeugtu	- Rp-1
Produk C	- Rp230000
Produk F	- Rp99999

## 3. Tampilan Edit Produk dengan Modal Pop-up

The screenshot shows a modal dialog box titled "Edit Produk". It contains two input fields: "Nama Produk: hrugeugtu" and "Harga: -1". Below the fields are two buttons: "Simpan" and "Batal". The background of the modal is semi-transparent.

## 4. Tampilan Hapus Produk dengan Notifikasi Konfirmasi

The screenshot shows a confirmation dialog box with the message "localhost:5173 menyatakan Apakah Anda yakin ingin menghapus produk ini?". It has "Oke" and "Batal" buttons. The background of the dialog is semi-transparent. The underlying page shows the same e-commerce application interface as in the previous screenshots, with the "Daftar Produk" table and a "Tambah Produk" form.

## 5. Tampilan Database di pgAdmin setelah CRUD

The screenshot shows the pgAdmin 4 interface. At the top, there are tabs for 'File', 'Object', 'Tools', 'Edit', 'View', 'Window', and 'Help'. Below the tabs, there are two connection tabs: 'ecommerce-kecil...' and 'ecommerce-kecil/postgres@PostgreSQL 17\*'. The second tab is selected. The main area has a toolbar with various icons for database management. Below the toolbar, there are two tabs: 'Query' (selected) and 'Query History'. In the 'Query' tab, the following SQL code is written:

```
1 select *from produk;
```

Below the query, the 'Data Output' tab is selected, showing a table with the results of the query. The table has four columns: 'id' [PK] integer, 'nama' character varying (50), and 'harga' integer. The data is as follows:

	id [PK] integer	nama character varying (50)	harga integer
1	38	Produk A	10000
2	39	Produk B	20000
3	41	Produk d	5000000
4	42	Produk E	300000
5	43	hrugeugtu	-1
6	47	hrugeugtu	-1
7	44	Produk C	230000
8	48	Produk F	99999

## 6. Hasil API Request di Postman

The screenshot shows a browser window with the address bar containing 'localhost:3001'. Below the address bar, there are links for 'Gmail', 'YouTube', 'Maps', and 'Adobe Acrobat'. The main content area displays the text 'API Berjalan' followed by a small rocket ship emoji.

← → ⌂

localhost:3001/api/produk



Gmail

YouTube

Maps

Adobe Acrobat

Pretty print

```
},
{
  "id": 42,
  "nama": "Produk E",
  "harga": 300000
},
{
  "id": 43,
  "nama": "hrugeugtu",
  "harga": -1
},
{
  "id": 47,
  "nama": "hrugeugtu",
  "harga": -1
},
{
  "id": 44,
  "nama": "Produk C",
  "harga": 230000
},
{
  "id": 48,
  "nama": "Produk F",
  "harga": 99999
},
```

- GET

The screenshot shows the Postman application interface. On the left, the 'EXPLORER' sidebar lists files and folders for 'ECOMMERCE-KECIL' and 'Yosan-frontend'. In the center, a 'GET' request is being made to 'http://localhost:3001/api/produk'. The 'Body' tab of the request configuration is selected. The response status is '200 OK', the time is '175 ms', and the size is '606 B'. The response body is displayed in 'Pretty' format, showing a JSON array of products:

```
[{"id": 44, "nama": "Produk C", "harga": 230000}, {"id": 48, "nama": "Produk F", "harga": 99999}]
```

- POST

- PUT

- DELETE

The screenshot shows the Postman application interface. At the top, there's a search bar with the text "ecommerce-kecil". Below it, a list of requests includes one for "index.js" and another for "produkR...". A new request is being built for "http://localhost:3001/api/produk/3" using the "DELETE" method. The "Body" tab is selected, showing a raw JSON payload:

```
1 {  
2   "nama": "Produk S",  
3   "harga": 850000  
4 }  
5
```

Below the body, the "Body" tab is active, followed by "Cookies", "Headers (8)", and "Test Results". Under "Body", there are buttons for "Pretty", "Raw", "Preview", and "JSON". The "Pretty" button is selected. The response body is displayed as:

```
1 {  
2   "message": "Produk dihapus"  
3 }
```

## 🔧 Troubleshooting

### 1. Node.js Tidak Berjalan



Gmail

YouTube

Maps

Adobe Acrobat

Cannot GET /

```
node -v
```

Pastikan Node.js sudah terinstall dengan benar.

## 1. Node.js Tidak Berjalan

The screenshot shows a POST request to `http://localhost:3001/api/produk`. The 'Body' tab is selected, showing a JSON payload:

```
1 {
2   "nama": "Produk 0",
3   "harga": 50000
4 }
```

The response status is `200 OK`, and the JSON response is:

```
1 [
2   {
3     "id": 55,
4     "nama": null,
5     "harga": null
6   }
]
```

## 1. Node.js Tidak Berjalan

The screenshot shows the pgAdmin 4 interface. In the top navigation bar, the 'File' menu is highlighted. Below the menu, there are two tabs: 'Welcome' and 'ecommerce-kecil/postgres@PostgreSQL 17\*'. The 'ecommerce-kecil/postgres@PostgreSQL 17\*' tab is active. The main area displays a query editor with the following SQL code:

```
1 SELECT column_name, data_type FROM information_schema.columns WHERE table_name = 'produk';
```

A tooltip 'Execute script F5' is shown over the F5 key icon. The results of the query are displayed in a table below:

	column_name	data_type
1	id	integer
2	nama	character varying
3	harga	integer

## 1. Node.js Tidak Berjalan

The screenshot shows the pgAdmin 4 interface. In the top navigation bar, the 'File' menu is highlighted. Below the menu, there are two tabs: 'Welcome' and 'ecommerce-kecil/postgres@PostgreSQL 17'. The 'ecommerce-kecil/postgres@PostgreSQL 17' tab is active. The main area displays a query editor with the following SQL code:

```
1 SELECT * FROM produk;
```

The results of the query are displayed in a table below:

	id	nama	harga
1	1	laptop	2000000
2	2	monitor	500000
3	3	mouse	50000
4	4	keyboar	100000
5	5	headset	150000
6	6	speaker	80000
7	7	charger	20000
8	8	usb drive	50000
9	9	power bank	100000
10	10	smartphone	3000000
11	11	tablet	1500000
12	12	camera	2500000
13	13	projector	1000000
14	14	scanner	500000
15	15	printer	150000
16	16	hard disk	80000
17	17	ram	50000
18	18	cpu	1000000
19	19	gpu	2000000
20	20	ssd	100000

Data Output   Messages   Notifications

**SQL**

	<b>id</b> [PK] integer	<b>nama</b> character varying (50)	<b>harga</b> integer
2	39	Produk D	200000
3	41	Produk d	5000000
4	42	Produk E	300000
5	43	hrugeugtu	-1
6	47	hrugeugtu	-1
7	44	Produk C	230000
8	48	Produk F	99999
9	49	[null]	[null]
10	50	[null]	[null]
11	51	[null]	[null]
12	52	[null]	[null]
13	53	[null]	[null]
14	54	[null]	[null]
15	55	[null]	[null]

Total rows: 15   Query complete 00:00:00.056

## 1. Node.js Tidak Berjalan

A screenshot of a web browser window. The address bar shows 'localhost:3001/api/produk'. Below the address bar is a toolbar with icons for Gmail, YouTube, Maps, and Adobe Acrobat. A 'Pretty print' button is visible on the left. The main content area displays the JSON error message: {"error": "Server error"}.

{"error": "Server error"}

### 1. Node.js Tidak Berjalan



### 2. Database PostgreSQL Tidak Terhubung

- Pastikan PostgreSQL sudah berjalan dan user `postgres` memiliki akses.
- Cek konfigurasi di `db.js`.
- Jalankan perintah berikut di PostgreSQL untuk memastikan database sudah ada:

```
SELECT * FROM produk;
```

### 3. Masalah Fetch Data di Frontend

- Pastikan backend berjalan dengan benar di `http://localhost:3001`.
- Gunakan `npm install axios` jika Axios belum terinstall.
- Pastikan CORS sudah diaktifkan di backend.

## ❖ Best Practices dalam Pengembangan

- **Gunakan Environment Variables:** Simpan kredensial database di `.env`.
  - **Gunakan GitHub Issues & Projects:** Kelola tugas dengan jelas.
  - **Gunakan Middleware di Express:** Untuk menangani error dan autentikasi.
  - **Optimasi Query di PostgreSQL:** Gunakan indeks jika perlu untuk performa tinggi.
  - **Pastikan UI Responsif:** Gunakan media queries dengan Tailwind CSS.
- 

## ❖ Commit dan Push ke GitHub

```
git add .
git commit -m "Menambahkan dokumentasi dan fitur baru"
git push origin main
```

### Inspirasi Hari Ini

"Mengetahui saja tidak cukup; kita harus menerapkannya. Berkeinginan saja tidak cukup; kita harus melakukannya." – Goethe