

# UNIVERSITAS GUNADARMA



## PRAKTIKUM PEMROGRAMAN WEB

### MANUAL BOOK

#### **“Aplikasi Pemograman Website Virtual Storage Menggunakan Laravel, React”**

Nama Anggota:

1. Gilberto Patrick Lie (50422622)
2. Muhammad Tarmidzi Bariq (51422161)
3. Muhammad Farhan Zidan (51422039)

Kelas: 3IA11

Fakultas: Teknik Industri

Jurusan: Teknik Informatika

Ketua Asisten: Fadhil Muhammad

**Ditulis Guna Melengkapi Sebagai Syarat Praktikum Pemrograman Web**

**jenjang S1**

**LABORATORIUM INFORMATIKA**

**UNIVERSITAS GUNADARMA**

**2025**

## **KATA PENGANTAR**

Puji syukur kehadiran Tuhan Yang Maha Esa atas segala karunia nikmatnya sehingga Manual Book yang berjudul (Aplikasi Pemrograman Website Virtual Storage Menggunakan Laravel, React) ini dapat diselesaikan dengan maksimal, tanpa ada halangan yang berarti. Makalah ini disusun untuk memenuhi Praktikum Pemrograman Web yang dibimbing oleh PJ yaitu Fadhil Muhammad

Kami menyadari bahwa penelitian ini masih jauh dari sempurna. Oleh karena itu, kami mengharapkan kritik dan saran yang membangun dari berbagai pihak untuk perbaikan di masa mendatang. Semoga penelitian ini dapat memberikan manfaat bagi perkembangan ilmu pengetahuan. Akhir kata, kami mengucapkan terima kasih kepada semua pihak yang telah membantu dalam penyelesaian penelitian ini.

Depok, 15 Januari 2025

Penulis

## DAFTAR ISI

<b>KATA PENGANTAR.....</b>	<b>2</b>
<b>DAFTAR ISI.....</b>	<b>3</b>
<b>BAB I</b>	
<b>PENDAHULUAN.....</b>	<b>4</b>
1.1 Latar Belakang.....	4
1.2 Tujuan.....	4
<b>BAB II</b>	
<b>PEMBAHASAN.....</b>	<b>5</b>
2.1 PHP Laravel.....	5
2.2 React JS.....	5
2.3 MySQL.....	6
2.4 Tailwind CSS.....	6
2.5 Vite JS.....	6
<b>BAB III</b>	
<b>LOGIKA PROGRAM.....</b>	<b>7</b>
3.1 Membuat Proyek Laravel.....	7
3.2 Cara Install dan Menjalankan React (Vite).....	12
3.3 Install dan Konfigurasi React Router DOM.....	13
3.4 Menampilkan Data di React dari Rest API.....	14
3.5 Insert dan Upload Data di React dengan Rest API.....	15
3.6 Edit dan Update Data di React dengan Rest API.....	18
3.7 Delete Data di React Menggunakan Rest API.....	19
<b>BAB IV.....</b>	<b>22</b>
<b>PENUTUP.....</b>	<b>22</b>
4.1 Kesimpulan.....	22
4.2 Saran.....	22

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Dalam pengembangan aplikasi web modern, React telah menjadi salah satu pustaka JavaScript yang paling populer karena kemampuannya dalam membangun antarmuka pengguna yang dinamis dan efisien. Namun, pengembang sering kali menghadapi tantangan dalam mengintegrasikan React dengan teknologi lain seperti Rest API untuk pengelolaan data. Oleh karena itu, diperlukan panduan praktis yang membahas langkah-langkah membangun aplikasi React secara bertahap, mulai dari instalasi, konfigurasi, hingga integrasi dengan Rest API. Artikel ini disusun untuk memberikan pembaca pemahaman yang komprehensif tentang pengembangan aplikasi React, sekaligus mempersiapkan mereka untuk mengatasi tantangan dalam proyek nyata.

### **1.2 Tujuan**

Tujuan dari aplikasi Pemrograman Website Virtual Storage Menggunakan Laravel dan React ini adalah untuk menyediakan platform penyimpanan file yang aman dan efisien berbasis web, yang memungkinkan pengguna untuk mengunggah, menyimpan, mengelola, dan mengakses file secara mudah melalui antarmuka yang interaktif. Aplikasi ini bertujuan untuk meningkatkan kenyamanan dan produktivitas pengguna dalam melakukan manajemen data, dengan fitur-fitur seperti pengelolaan produk, validasi data yang ketat, serta integrasi antara front-end dan back-end menggunakan teknologi Laravel dan React. Selain itu, aplikasi ini juga bertujuan untuk memberikan pengalaman pengguna yang responsif dan ramah pengguna, serta memastikan keamanan dan ketersediaan data secara maksimal.

## **BAB II**

### **PEMBAHASAN**

#### **2.1 PHP Laravel**

PHP Laravel adalah salah satu framework pengembangan aplikasi web yang populer berbasis bahasa pemrograman. PHP Laravel dirancang untuk memudahkan pembuatan aplikasi web modern. Dengan mengadopsi pola arsitektur MVC (Model-View-Controller), Laravel membantu pengembang menulis kode yang terstruktur dan terorganisir. Framework ini menawarkan berbagai fitur canggih, seperti routing yang fleksibel, migrasi basis data yang mudah dikelola, sistem autentikasi bawaan, dan integrasi dengan ORM Eloquent untuk mengelola database. Laravel juga memiliki ekosistem yang luas, termasuk alat seperti Artisan untuk otomatisasi tugas, Blade sebagai engine template, dan layanan seperti Laravel Forge untuk manajemen server. Kombinasi fitur tersebut membuat Laravel menjadi pilihan populer bagi pengembang yang menginginkan efisiensi dan kemudahan dalam mengembangkan aplikasi web.

#### **2.2 React JS**

React.js adalah pustaka JavaScript open-source yang dirancang untuk membangun antarmuka pengguna (UI) yang responsif dan dinamis, khususnya pada aplikasi web dengan banyak interaksi. React berfokus pada konsep komponen, yaitu blok kode yang dapat digunakan kembali untuk menciptakan elemen UI. Dengan memanfaatkan Virtual DOM, React mempercepat proses rendering dan meningkatkan performa aplikasi, terutama saat menangani perubahan data secara real-time. React menggunakan pendekatan deklaratif, di mana pengembang cukup mendefinisikan bagaimana UI seharusnya terlihat berdasarkan state data, dan React akan menangani pembaruan secara efisien. Pustaka ini sering digunakan dalam pengembangan aplikasi modern karena fleksibilitasnya serta dukungan ekosistemnya yang luas, seperti React Router untuk navigasi dan Redux untuk manajemen state.

## 2.3 MySQL

MySQL adalah sistem manajemen basis data relasional (RDBMS) open-source yang populer, dirancang untuk menyimpan, mengelola, dan mengolah data dengan efisien. MySQL menggunakan Structured Query Language (SQL) sebagai bahasa utama untuk berinteraksi dengan database. MySQL cocok digunakan untuk berbagai skala aplikasi, mulai dari proyek kecil hingga sistem besar dengan data dalam jumlah masif. Dengan arsitektur client-server, MySQL dapat diakses dari berbagai platform dan mendukung berbagai fitur seperti replikasi data, transaksi, dan keamanan data yang andal. Sistem ini sering digunakan sebagai backend database untuk aplikasi web, terutama dalam ekosistem LAMP (Linux, Apache, MySQL, PHP/Python/Perl), karena kecepatan, skalabilitas, dan keandalannya.

## 2.4 Tailwind CSS

Tailwind CSS adalah framework CSS yang membantu pengembang mendesain tampilan website dengan cepat dan mudah. Tailwind menyediakan kelas-kelas siap pakai untuk mengatur hal-hal seperti warna, ukuran, jarak, atau posisi elemen. Jadi, cukup menambahkan nama kelas langsung ke tag HTML untuk mengatur tampilannya. Dengan pendekatan ini, kita bisa membuat desain yang rapi tanpa perlu membuat file CSS yang panjang dan rumit. Tailwind fleksibel karena kita bisa menyesuaikan temanya sesuai kebutuhan, dan fitur JIT (Just-in-Time) memastikan hanya kode yang dipakai saja yang masuk ke file akhir, sehingga desain lebih ringan dan cepat.

## 2.5 Vite JS

Vite adalah sebuah framework JavaScript open-source yang digunakan untuk membangun aplikasi frontend yang cepat dan efisien. Vite berfokus pada performa yang cepat dan pengembangan yang mudah. Salah satu fitur unggulan dari Vite adalah kemampuannya untuk melakukan hot-reloading, yang memungkinkan pengembang untuk melihat perubahan pada kode secara langsung tanpa perlu melakukan reload pada halaman web.

## BAB III

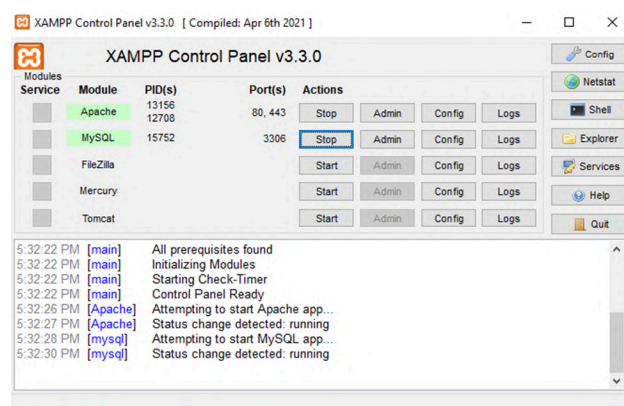
### LOGIKA PROGRAM

Pada kesempatan kali ini kita akan menjelaskan bagaimana cara membuat CRUD menggunakan React.js dan Laravel 10. React.js akan kita gunakan sebagai frontend yang mengkonsumsi atau mengambil data dari Rest API yang dibuat oleh Laravel. Kita juga akan belajar menggunakan build tools yang bernama Vite untuk membuat project React.js-nya.

#### 3.1 Membuat Proyek Laravel

- Langkah pertama:

PHP versi terbaru yang kompatibel dengan Laravel. Composer, yaitu manajer dependensi PHP yang digunakan oleh Laravel Database (seperti MySQL).



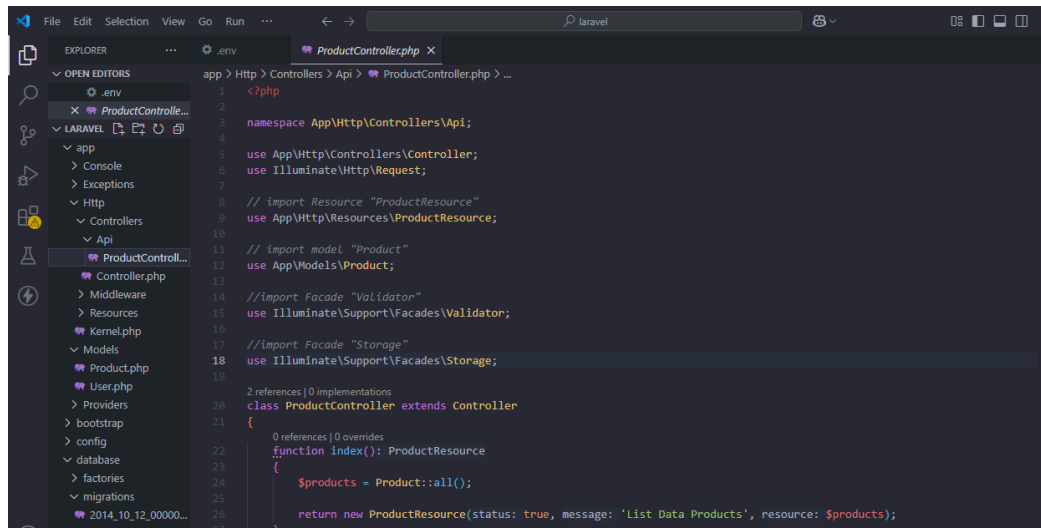
```
LENOVO@DESKTOP-RHU7QCV MINGW64 ~/Downloads/ujian_akhir-main
$ php -v
PHP 8.2.12 (cli) (built: Oct 24 2023 21:15:15) (ZTS Visual C++ 2019 x64)
Copyright (c) The PHP Group
Zend Engine v4.2.12, Copyright (c) Zend Technologies

LENOVO@DESKTOP-RHU7QCV MINGW64 ~/Downloads/ujian_akhir-main
$ composer --version
Composer version 2.8.4 2024-12-11 11:57:47
PHP version 8.2.12 (C:\xampp1\php\php.exe)
Run the "diagnose" command to get more detailed diagnostics output.
```

Mulai dari proyek baru, gunakan perintah **composer create-project** untuk membuat proyek Laravel baru. Misalnya:

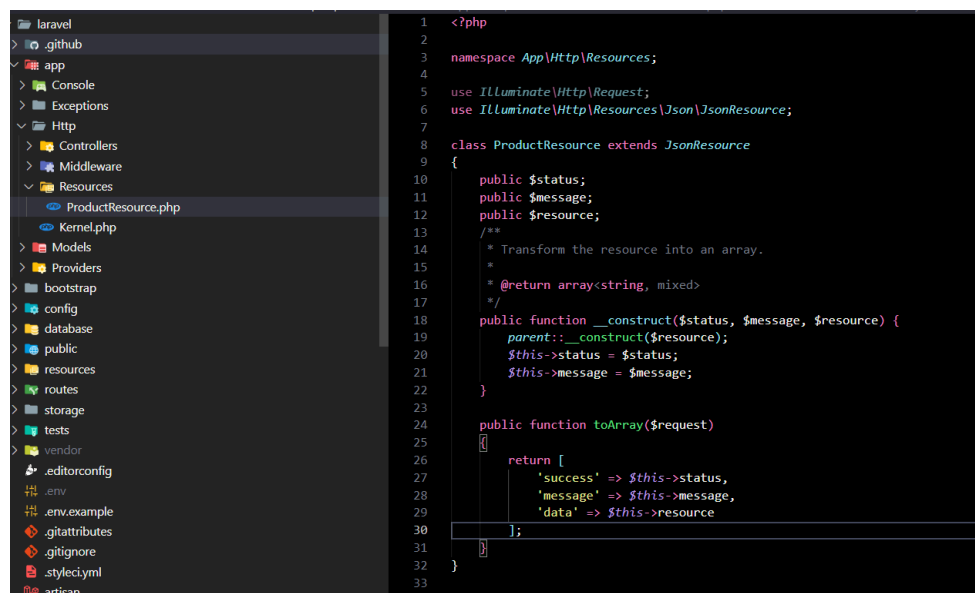
```
composer create-project --prefer-dist laravel/laravel
```

Berikutnya buat file di App/Http/Controller/Api/ProductController.php. Controller API di Laravel yang menangani operasi CRUD untuk produk, termasuk mengambil daftar produk, menambah, memperbarui, dan menghapus produk, serta mengelola unggahan gambar produk.



```
1 <?php
2
3 namespace App\Http\Controllers\Api;
4
5 use App\Http\Controllers\Controller;
6 use Illuminate\Http\Request;
7
8 // import Resource "ProductResource"
9 use App\Http\Resources\ProductResource;
10
11 // import model "Product"
12 use App\Models\Product;
13
14 //import Facade "Validator"
15 use Illuminate\Support\Facades\Validator;
16
17 //import Facade "Storage"
18 use Illuminate\Support\Facades\Storage;
19
20 2 references | 0 implementations
21 class ProductController extends Controller
22 {
23     0 references | 0 overrides
24     function index(): ProductResource
25     {
26         $products = Product::all();
27
28         return new ProductResource(status: true, message: 'List Data Products', resource: $products);
29     }
30 }
```

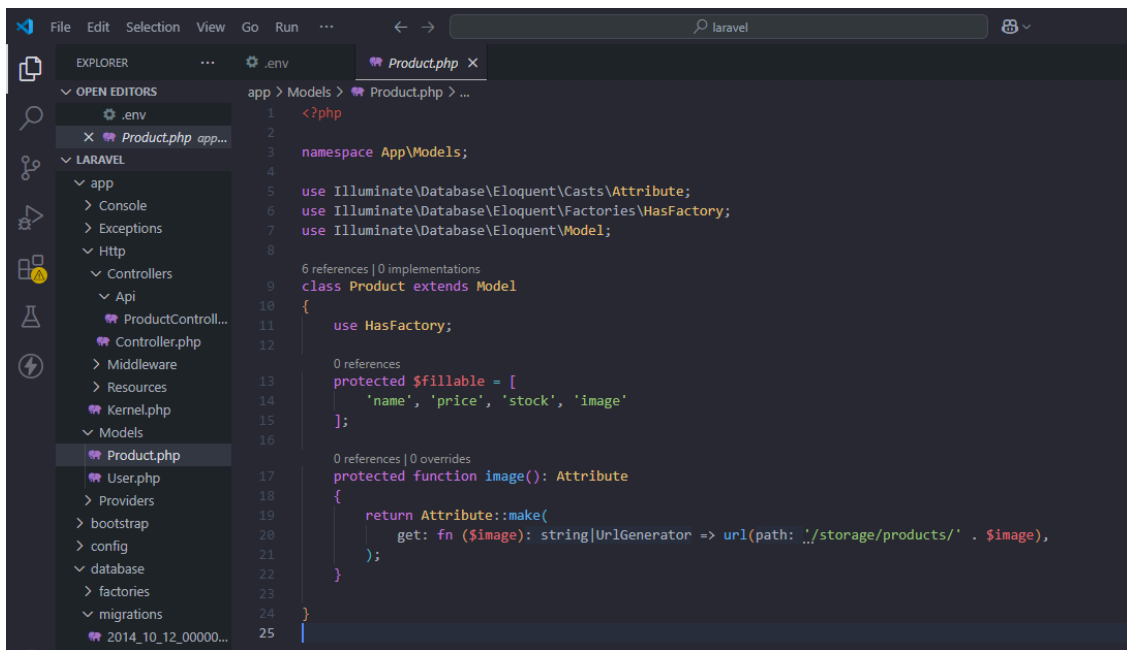
Berikutnya buat file di App/Http/Resource/ProductResource.php. Validasi input dan respons yang terstruktur menggunakan ProductResource.



```
1 <?php
2
3 namespace App\Http\Resources;
4
5 use Illuminate\Http\Request;
6 use Illuminate\Http\Resources\Json\JsonResource;
7
8 class ProductResource extends JsonResource
9 {
10     public $status;
11     public $message;
12     public $resource;
13
14     /**
15      * Transform the resource into an array.
16      *
17      * @return array<string, mixed>
18      */
19     public function __construct($status, $message, $resource) {
20         parent::__construct($resource);
21         $this->status = $status;
22         $this->message = $message;
23     }
24
25     public function toArray($request)
26     {
27         return [
28             'success' => $this->status,
29             'message' => $this->message,
30             'data' => $this->resource
31         ];
32     }
33 }
```

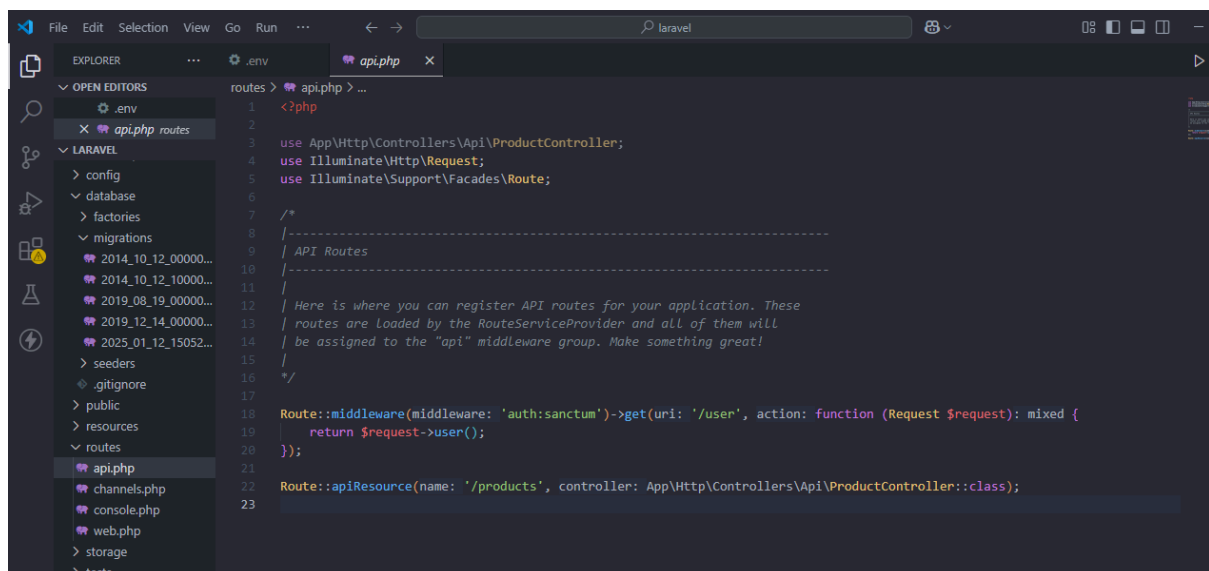
Selanjutnya kita buat folder di Model/Product.php, mendefinisikan model Product di Laravel yang menggunakan fitur HasFactory untuk factory dan memiliki atribut yang dapat diisi (fillable), serta mengimplementasikan accessor untuk atribut image yang mengembalikan URL gambar produk dari direktori penyimpanan.





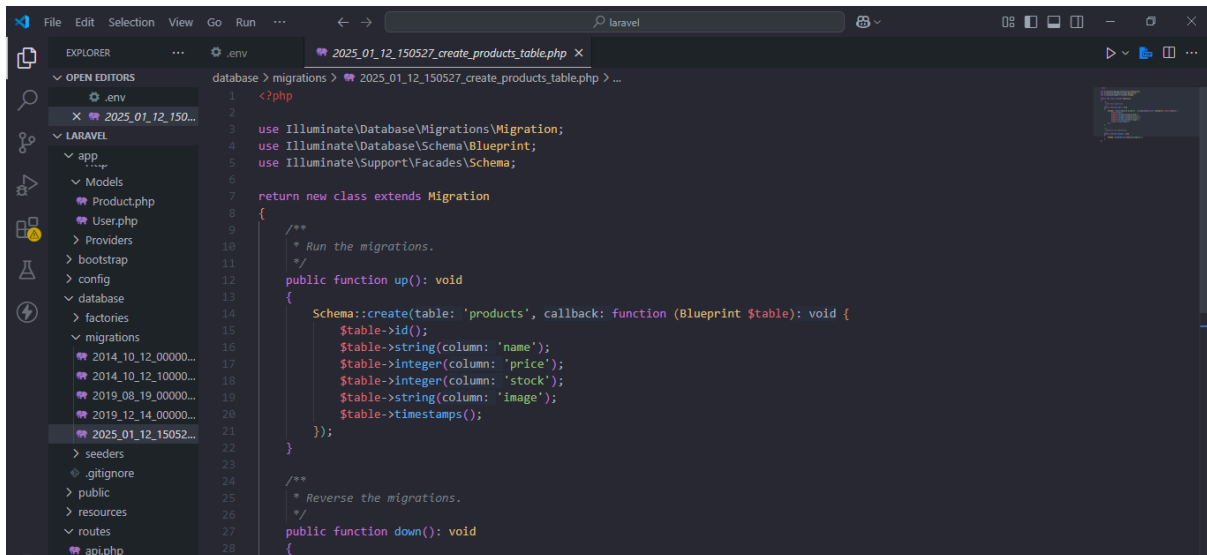
```
1 <?php
2
3 namespace App\Models;
4
5 use Illuminate\Database\Eloquent\Casts\Attribute;
6 use Illuminate\Database\Eloquent\Factories\HasFactory;
7 use Illuminate\Database\Eloquent\Model;
8
9 class Product extends Model
10 {
11     use HasFactory;
12
13     protected $fillable = [
14         'name', 'price', 'stock', 'image'
15     ];
16
17     protected function image(): Attribute
18     {
19         return Attribute::make(
20             get: fn ($image): string|UrlGenerator => url(path: './storage/products/' . $image),
21         );
22     }
23 }
24
25
```

Berikutnya kita akan buat kembali folder di bagian routes/api.php, mendefinisikan rute API di Laravel. Rute pertama menggunakan middleware auth:sanctum untuk mengambil data pengguna yang terautentikasi. Rute kedua mendefinisikan sumber daya API untuk ProductController dengan menggunakan Route::apiResource, yang secara otomatis membuat rute untuk operasi CRUD (create, read, update, delete) pada produk.



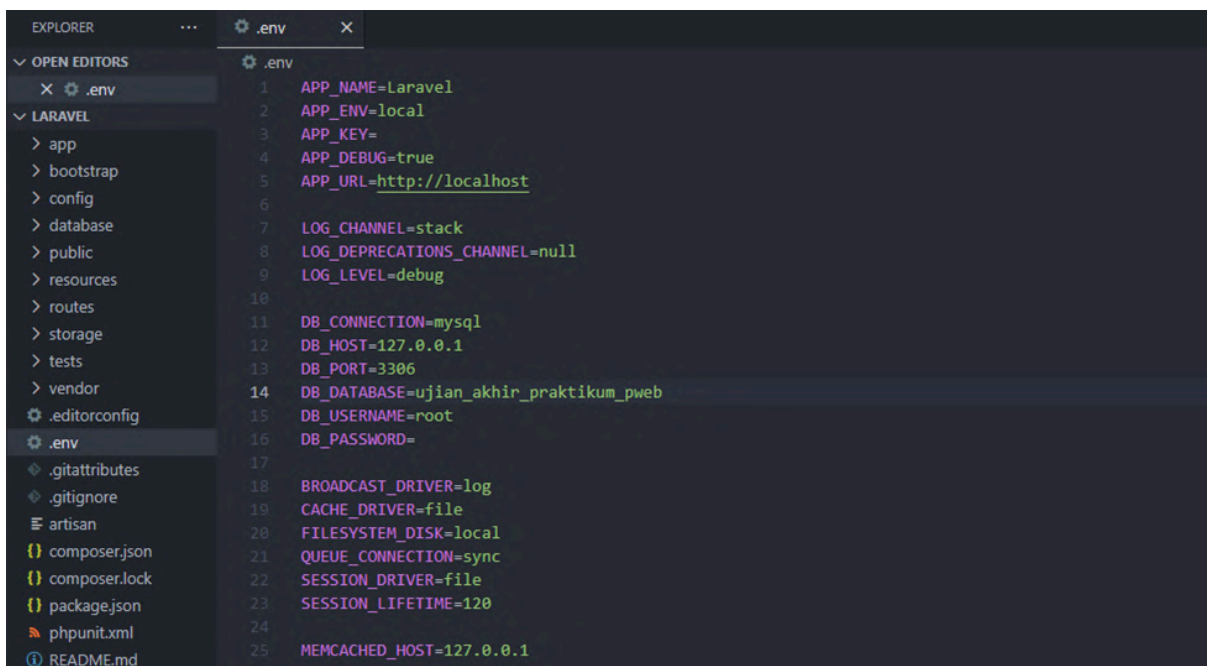
```
1 <?php
2
3 use App\Http\Controllers\Api\ProductController;
4 use Illuminate\Http\Request;
5 use Illuminate\Support\Facades\Route;
6
7 /*
8 |-----
9 | API Routes
10 |-----
11 |
12 | Here is where you can register API routes for your application. These
13 | routes are loaded by the RouteServiceProvider and all of them will
14 | be assigned to the "api" middleware group. Make something great!
15 |
16 */
17
18 Route::middleware(middleware: 'auth:sanctum')->get(uri: '/user', action: function (Request $request): mixed {
19     return $request->user();
20 });
21
22 Route::apiResource(name: '/products', controller: App\Http\Controllers\Api\ProductController::class);
23
```

Lalu kita masuk ke dalam file migration, sebuah migrasi untuk membuat tabel products dalam database menggunakan Laravel. Pada metode up(), tabel products akan dibuat dengan kolom-kolom seperti id, name, price, stock, image, dan timestamps. Sedangkan pada metode down(), migrasi ini akan menghapus tabel products jika ada perubahan yang perlu dibatalkan.



```
1 <?php
2
3 use Illuminate\Database\Migrations\Migration;
4 use Illuminate\Database\Schema\Blueprint;
5 use Illuminate\Support\Facades\Schema;
6
7 return new class extends Migration
8 {
9     /**
10      * Run the migrations.
11      */
12     public function up(): void
13     {
14         Schema::create(table: 'products', callback: function (Blueprint $table): void {
15             $table->id();
16             $table->string(column: 'name');
17             $table->integer(column: 'price');
18             $table->integer(column: 'stock');
19             $table->string(column: 'image');
20             $table->timestamps();
21         });
22     }
23
24     /**
25      * Reverse the migrations.
26      */
27     public function down(): void
28     {
29     }
```

Selanjutnya buka file .env.example ubah(rename)menjadi .env dan sesuaikan nama database yang nantinya ingin dibuat mysql seperti (DB\_DATABASE=ujian\_akhir\_praktikum\_pweb)



```
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_DEPRECATIONS_CHANNEL=null
9 LOG_LEVEL=debug
10
11 DB_CONNECTION=mysql
12 DB_HOST=127.0.0.1
13 DB_PORT=3306
14 DB_DATABASE=ujian_akhir_praktikum_pweb
15 DB_USERNAME=root
16 DB_PASSWORD=
17
18 BROADCAST_DRIVER=log
19 CACHE_DRIVER=file
20 FILESYSTEM_DISK=local
21 QUEUE_CONNECTION=sync
22 SESSION_DRIVER=file
23 SESSION_LIFETIME=120
24
25 MEMCACHED_HOST=127.0.0.1
```

Next, buka phpmyadmin buat database **ujian\_akhir\_praktikum\_pweb**, nanti akan dibuat secara otomatis untuk isi tabelnya.

Server: 127.0.0.1 > Database: ujian\_akhir\_praktikum\_pweb

Struktur SQL Cari Kueri Ekspor Impor Operasi Hak Akses Routine Event Trigger Pelacakan

Filters

Mengandung kata:

Tabel	Tindakan	Baris	Jenis	Penyortiran	Ukuran	Beban
<input type="checkbox"/> failed_jobs	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
<input type="checkbox"/> migrations	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	5	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
<input type="checkbox"/> password_reset_tokens	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
<input type="checkbox"/> personal_access_tokens	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	48.0 KB	-
<input type="checkbox"/> products	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	2	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
<input type="checkbox"/> users	★ Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
<b>6 tabel</b>	<b>Jumlah</b>	<b>7</b>	<b>InnoDB</b>	<b>utf8mb4_general_ci</b>	<b>160.0 KB</b>	<b>0 B</b>

☐ Pilih Semua
 Dengan pilihan:

- langkah kedua Install Dependensi

Jika sudah memiliki proyek Laravel dan ingin menginstall dependensi yang diperlukan, cukup jalankan di terminal: `composer install`

```
PS C:\Users\LENOVO\Downloads\ujian_akhir-main\laravel> composer install
Installing dependencies from lock file (including require-dev)
Verifying lock file contents can be installed on current platform.
Package operations: 111 installs, 0 updates, 0 removals
- Installing doctrine/inflector (2.0.10): Extracting archive
- Installing doctrine/lexer (3.0.1): Extracting archive
- Installing symfony/polyfill-ctype (v1.31.0): Extracting archive
- Installing webmozart/assert (1.11.0): Extracting archive
- Installing dragonmantank/cron-expression (v3.4.0): Extracting archive
- Installing symfony/deprecation-contracts (v3.5.1): Extracting archive
- Installing psr/container (2.0.2): Extracting archive
- Installing fakerphp/faker (v1.24.1): Extracting archive
- Installing symfony/polyfill-php83 (v1.31.0): Extracting archive
- Installing symfony/polyfill-mbstring (v1.31.0): Extracting archive
```

Selanjutnya kita coba composer ketik di terminal **dump-autoload** (Ini akan memperbarui autoloader yang digunakan oleh Composer)

```
PS C:\Users\LENOVO\Downloads\ujian_akhir-main\laravel> composer dump-autoload
Generating optimized autoload files
> Illuminate\Foundation\ComposerScripts::postAutoloadDump
> @php artisan package:discover --ansi

INFO Discovering packages.

laravel/sail ..... DONE
laravel/sanctum ..... DONE
laravel/tinker ..... DONE
nesbot/carbon ..... DONE
nunomaduro/collision ..... DONE
nunomaduro/termwind ..... DONE
```

berikutnya migrate Database lalu ketik **php artisan migrate** Perintah ini akan menjalankan skrip migrasi yang ada untuk membuat atau memperbarui struktur database sesuai dengan kebutuhan aplikasi.

Lalu Jalankan Server Lokal Setelah semuanya siap dan migrasi telah selesai (atau jika tidak ada migrasi yang perlu dilakukan), kamu bisa menjalankan aplikasi Laravel di server lokal dengan perintah: **php artisan serve**

```
Generated optimized autoload files containing 6104 classes
PS C:\Users\LENOVO\Downloads\ujian_akhir-main\laravel> php artisan migrate

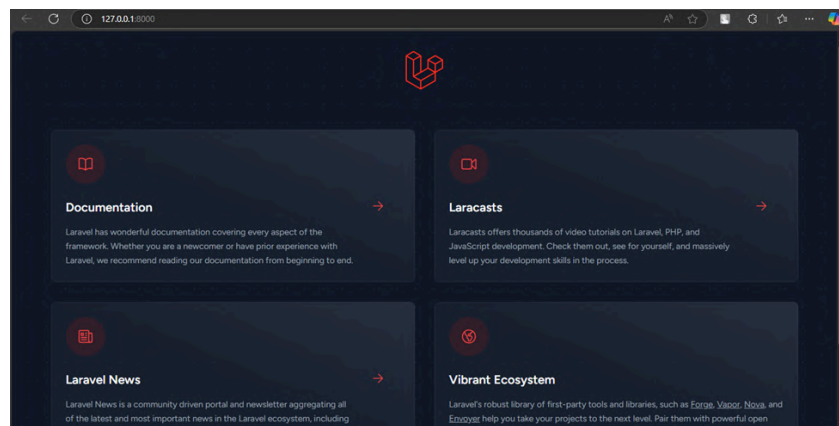
  INFO  Nothing to migrate.

PS C:\Users\LENOVO\Downloads\ujian_akhir-main\laravel> php artisan serve

  INFO  Server running on [http://127.0.0.1:8000].

Press Ctrl+C to stop the server
```

Server akan berjalan di <http://127.0.0.1:8000> (atau URL lain jika ada pengaturan berbeda). Ini akan membuat aplikasi Laravel dapat diakses melalui browser.



## 3.2 Cara Install dan Menjalankan React (Vite)

- Langkah pertama Instalasi Node.js:

Sebelum kita lanjutkan, kita harus menginstall Node.js terlebih dahulu di dalam komputer. Jika belum menginstall-nya, maka silahkan bisa menginstallnya dengan mengikuti link berikut ini <https://nodejs.org/en>, silahkan disesuaikan dengan sistem operasi yang digunakan. Untuk memeriksa apakah Node.js sudah berhasil terinstall, silahkan jalankan perintah berikut ini di dalam terminal/CMD.

```

LENOVO@DESKTOP-RHU7QCV MINGW64 ~/Downloads/ujian_akhir-main/react-crud
$ node --version
v22.13.0

LENOVO@DESKTOP-RHU7QCV MINGW64 ~/Downloads/ujian_akhir-main/react-crud
$ npm --version
10.9.2

```

- Langkah 2 Membuat Project React dengan Vite:

Setelah Node.js sudah berhasil terinstall, maka kita bisa lanjutkan membuat project React menggunakan Vite. Silahkan masuk ke dalam folder dimana akan menyimpan project-nya, kemudian jalankan perintah berikut ini di dalam terminal/CMD.

```

LENOVO@DESKTOP-RHU7QCV MINGW64 ~/Downloads/ujian_akhir-main/react-crud
$ npm create vite@4.2.0 react-crud

```

Jika perintah di atas berhasil dijalankan, maka kita akan berhasil dibuatkan scaffolding project-nya dengan nama react-crud. Setelah itu, silahkan jalankan perintah berikut ini untuk masuk ke dalam folder project-nya.

```

LENOVO@DESKTOP-RHU7QCV MINGW64 ~/Downloads/ujian_akhir-main/react-crud
$ npm install

```

Dan silahkan tunggu proses installasinya sampai selesai dan pastikan terhubung dengan internet.

- Langkah 3 Menjalankan Project React dengan Vite

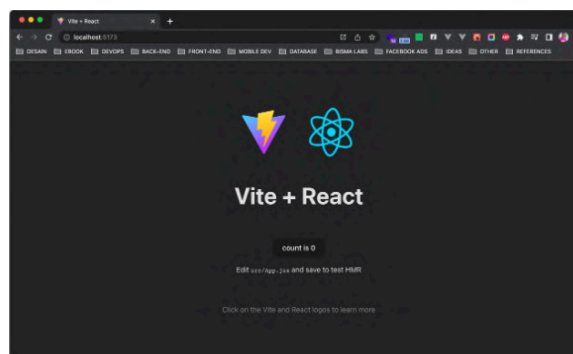
Setelah proses instalasi selesai, silahkan jalankan perintah berikut ini untuk menjalankan project React menggunakan Vite.

```

LENOVO@DESKTOP-RHU7QCV MINGW64 ~/Downloads/ujian_akhir-main/react-crud
$ npm run dev

```

Jika berhasil akan tampil seperti berikut:



### 3.3 Install dan Konfigurasi React Router DOM

React Router DOM adalah sebuah library yang digunakan dalam pengembangan aplikasi web dengan menggunakan React. Library ini memungkinkan untuk membuat navigasi pada aplikasi web menjadi cepat dan SPA atau Single Page Application.

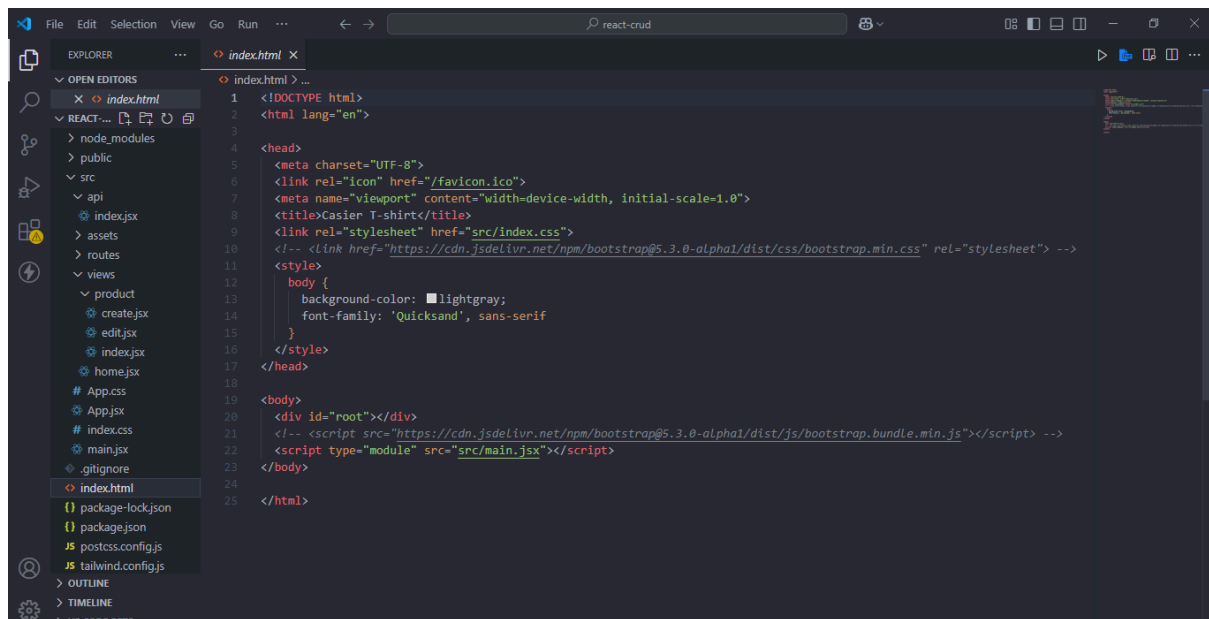
- Langkah 1 Installasi React Router DOM

Silahkan jalankan perintah berikut ini di dalam terminal/CMD dan pastikan sudah berada di dalam project React-nya.

```
LENOVO@DESKTOP-RHU7QCV MINGW64 ~/Downloads/ujian_akhir-main/react-crud
$ npm install react-router-dom@6.4.5
```

- Langkah 2 buat folder index.html

template halaman HTML untuk aplikasi web "Casier T-shirt" yang memuat elemen dasar seperti pengaturan favicon, judul halaman, pengaturan gaya CSS, dan tempat untuk memuat aplikasi React melalui elemen dengan id root.



### 3.4 Menampilkan Data di React dari Rest API

Pada kesempatan kali ini Kita akan menampilkan data di dalam React dari Rest API yang dibuat di Laravel.

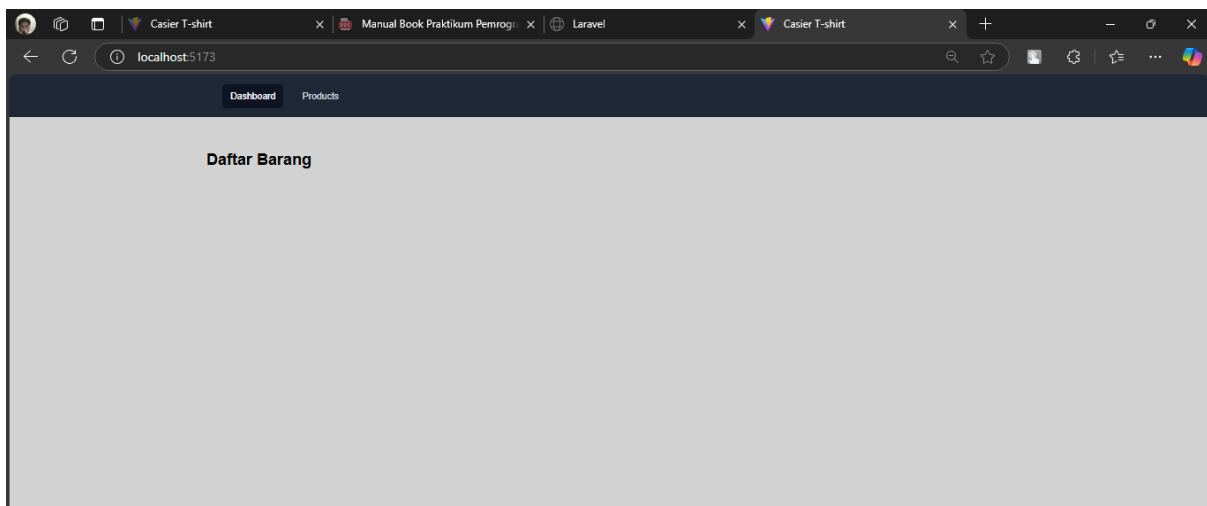
- Langkah 1 Installasi Axios

Langkah pertama yang harus kita lakukan adalah menginstall library tambahan yang bernama Axios. Library ini digunakan untuk melakukan Http request ke server dengan lebih mudah dan cepat. Silahkan jalankan perintah berikut ini di dalam terminal/CMD dan pastikan sudah berada di dalam project React-nya.

```
npm install axios@1.3.4
```

- Langkah 2 Uji Coba Menampilkan Data

Setelah semua berhasil dibuat, sekarang kita akan lakukan uji coba di dalam browser untuk melihat hasilnya, silahkan klik menu POSTS yang ada di navbar, atau bisa juga ke URL berikut ini <http://localhost:5173/>, jika berhasil maka akan menampilkan hasil seperti berikut.



### 3.5 Insert dan Upload Data di React dengan Rest API

Cara melakukan proses insert dan upload di dalam React menggunakan Rest API. Juga belajar bagaimana cara membuat proses insert data dan upload gambar di dalam React. Kita juga belajar menggunakan FormData untuk mengirim file ke dalam server.

- Langkah 1 - Insert dan Upload Data di React

Sekarang kita akan melakukan perubahan di dalam view post create, silahkan buka file `src/views/posts/create.jsx`, pertama kita import `useState` dari React. Kode ini adalah komponen React untuk halaman form pembuatan produk, yang memungkinkan pengguna

untuk memasukkan nama produk, harga, stok, dan gambar produk. Form ini dilengkapi dengan validasi untuk memastikan bahwa semua input valid, dan jika ada kesalahan, pesan kesalahan akan ditampilkan. Setelah form disubmit, data dikirim ke API untuk menyimpan produk baru, dan setelah berhasil, pengguna akan diarahkan ke halaman daftar produk.

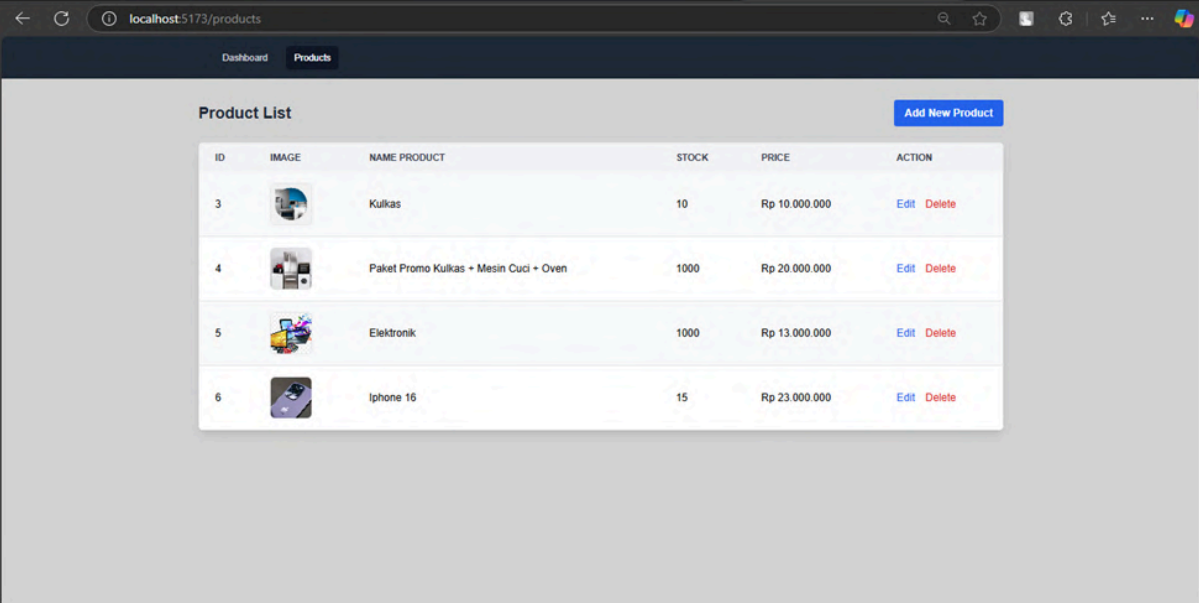
```
Go Run ... react-crud
create.jsx x
src > views > product > create.jsx > ProductCreate
1 // import useState, useEffect from 'react';
2 import { useState, useEffect } from 'react';
3
4 // import useNavigate
5 import { useNavigate } from 'react-router-dom';
6
7 // import api
8 import api from '../api';
9
10
11 export default function ProductCreate() {
12
13   // define state
14   const [image, setImage] = useState('');
15   const [name, setName] = useState('');
16   const [stock, setStock] = useState('');
17   const [price, setPrice] = useState('');
18
19   // state validation
20   const [errors, setErrors] = useState({});
21
22   const navigate = useNavigate();
23
24   const handleFileChange = (e) => {
25     setImage(e.target.files[0]);
26   }
27
28   const validateForm = () => {
29     const newErrors = {};
30     if (!image) newErrors.image = "Image is required";
31     if (!name) newErrors.name = "Name is required";
32     if (!stock || stock <= 0) newErrors.stock = "Stock must be greater than 0";
33     if (!price || price <= 0) newErrors.price = "Price must be greater than 0";
```

insert dan upload data, kita ke Product lalu Add New Product, masukkan beberapa data





The screenshot shows a web browser at the URL `localhost:5173/products/edit/4`. The page has a dark header with a 'Dashboard' link and a 'Products' link. The main content area features a product edit form. At the top of the form is a placeholder image of a kitchen appliance. Below the image is a 'Choose File' button and the text 'No file chosen'. The form contains four input fields: 'Name' with the value 'Paket Promo Kulkas • Mesin Cuci • Oven', 'Stock' with the value '1000', and 'Price' with the value '2000000'. An 'Update' button is located at the bottom of the form.



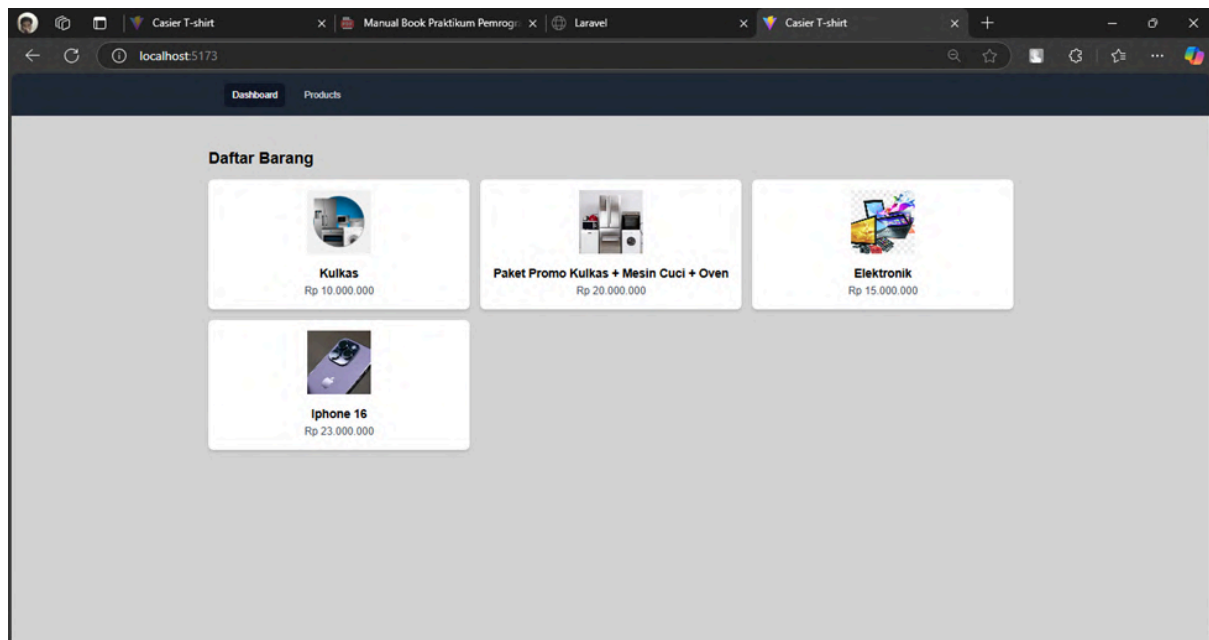
Akan menampilkan hasil data-data yang telah di input seperti berikut di bagian product:



The screenshot shows a web application interface for managing products. The page title is 'Product List' and there is an 'Add New Product' button. The table displays the following data:

ID	IMAGE	NAME PRODUCT	STOCK	PRICE	ACTION
3		Kulkas	10	Rp 10.000.000	<a href="#">Edit</a> <a href="#">Delete</a>
4		Paket Promo Kulkas + Mesin Cuci + Oven	1000	Rp 20.000.000	<a href="#">Edit</a> <a href="#">Delete</a>
5		Elektronik	1000	Rp 13.000.000	<a href="#">Edit</a> <a href="#">Delete</a>
6		Iphone 16	15	Rp 23.000.000	<a href="#">Edit</a> <a href="#">Delete</a>

dan dibagian Dashboard akan menampilkan semua jenis Produk seperti berikut:

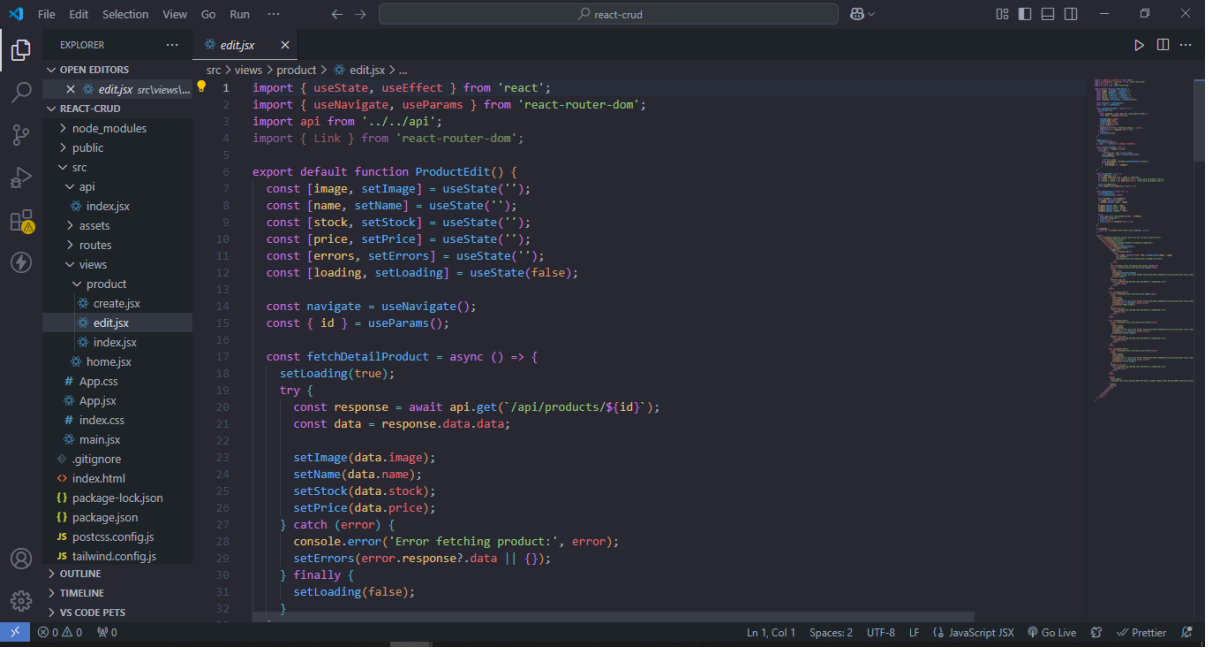


### 3.6 Edit dan Update Data di React dengan Rest API

Cara membuat proses edit dan update data di React menggunakan Rest API.

- Langkah 1 - Edit dan Update Data di React

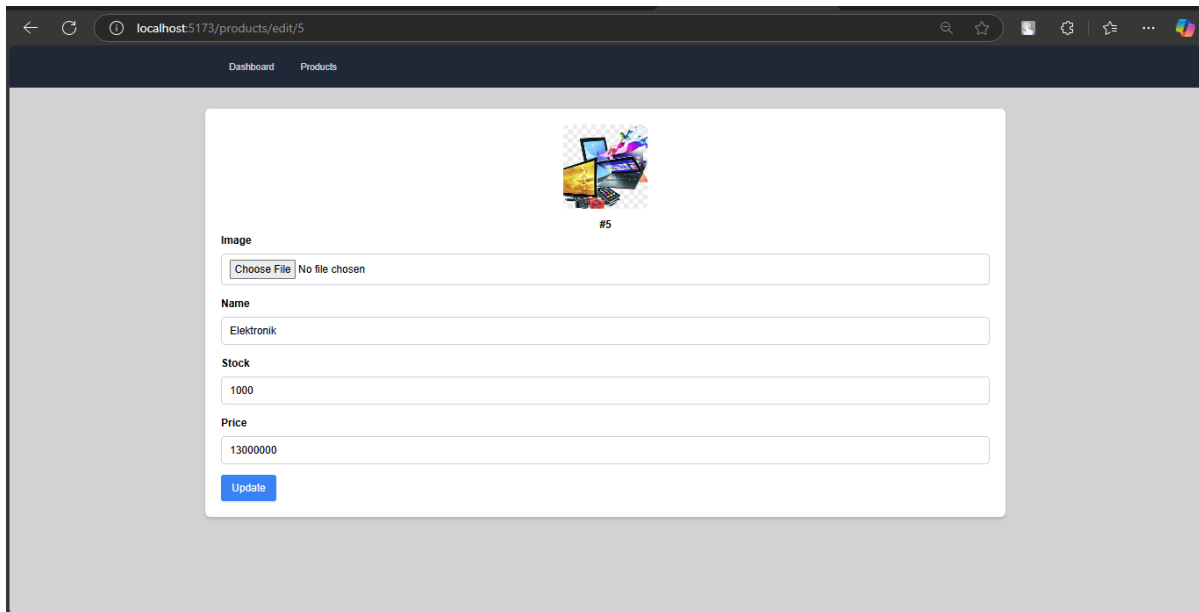
Silahkan buka file `src/views/posts/edit.jsx`. komponen React yang digunakan untuk mengedit data produk. Di dalam komponen ini, terdapat beberapa state untuk menyimpan data gambar, nama, stok, harga, dan error. Pada `useEffect`, data produk dengan ID tertentu akan diambil dari API dan ditampilkan dalam form. Ada juga validasi form untuk memastikan bahwa data yang dimasukkan sesuai, seperti nama, stok, dan harga yang harus lebih besar dari 0. Selain itu, jika ada gambar yang diunggah, gambar tersebut akan dipratinjau sebelum dikirimkan bersama dengan data lainnya melalui API untuk memperbarui produk. Jika form berhasil dikirim, pengguna akan diarahkan kembali ke halaman produk.



```
1 import { useState, useEffect } from 'react';
2 import { useNavigate, useParams } from 'react-router-dom';
3 import api from '../api';
4 import { Link } from 'react-router-dom';
5
6 export default function ProductEdit() {
7   const [image, setImage] = useState('');
8   const [name, setName] = useState('');
9   const [stock, setStock] = useState('');
10  const [price, setPrice] = useState('');
11  const [errors, setErrors] = useState('');
12  const [loading, setLoading] = useState(false);
13
14  const navigate = useNavigate();
15  const { id } = useParams();
16
17  const fetchDetailProduct = async () => {
18    setLoading(true);
19    try {
20      const response = await api.get(`/api/products/${id}`);
21      const data = response.data.data;
22
23      setImage(data.image);
24      setName(data.name);
25      setStock(data.stock);
26      setPrice(data.price);
27    } catch (error) {
28      console.error('Error fetching product:', error);
29      setErrors(error.response?.data || {});
30    } finally {
31      setLoading(false);
32    }
33  }
```

- Langkah 2 - Uji Coba Edit dan Update Data

klik button EDIT yang ada pada data yang dimiliki, jika berhasil maka akan menampilkan halaman form edit data seperti berikut. Misal kita mau edit untuk Nama barang Elektronik:



Kita akan mengedit bagian harga karena ada kenaikan menjadi 15.000.000 akan tampil seperti berikut dan diupdate:

The screenshot shows the 'Product List' page with the following data:

ID	IMAGE	NAME PRODUCT	STOCK	PRICE	ACTION
3	[Image icon]	Kulkas	10	Rp 10.000.000	<a href="#">Edit</a> <a href="#">Delete</a>
4	[Image icon]	Paket Promo Kulkas + Mesin Cuci + Oven	1000	Rp 20.000.000	<a href="#">Edit</a> <a href="#">Delete</a>
5	[Image icon]	Elektronik	1000	Rp 15.000.000	<a href="#">Edit</a> <a href="#">Delete</a>
6	[Image icon]	Iphone 16	15	Rp 23.000.000	<a href="#">Edit</a> <a href="#">Delete</a>

### 3.7 Delete Data di React Menggunakan Rest API

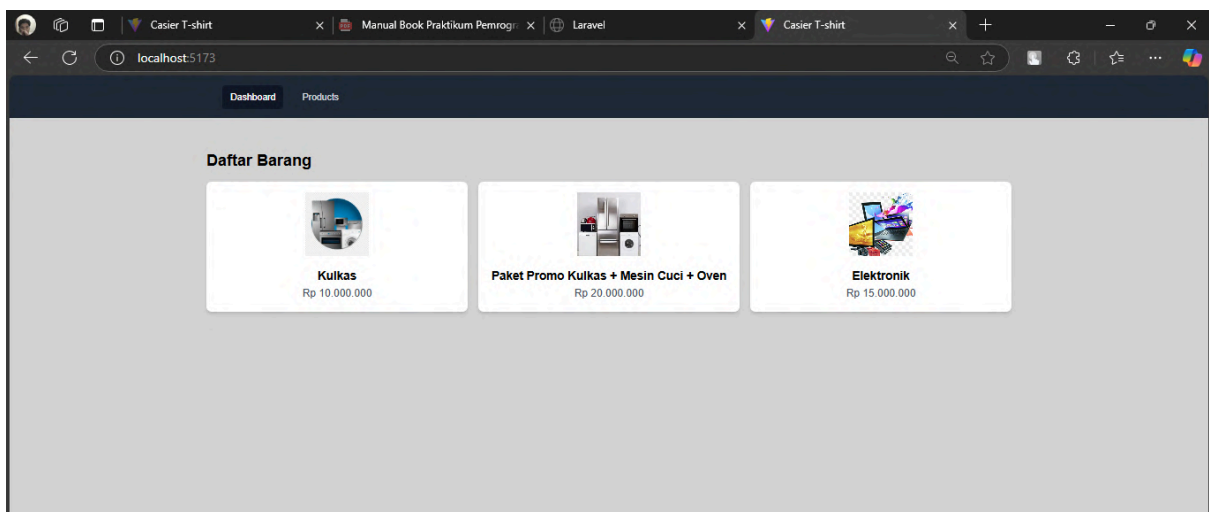
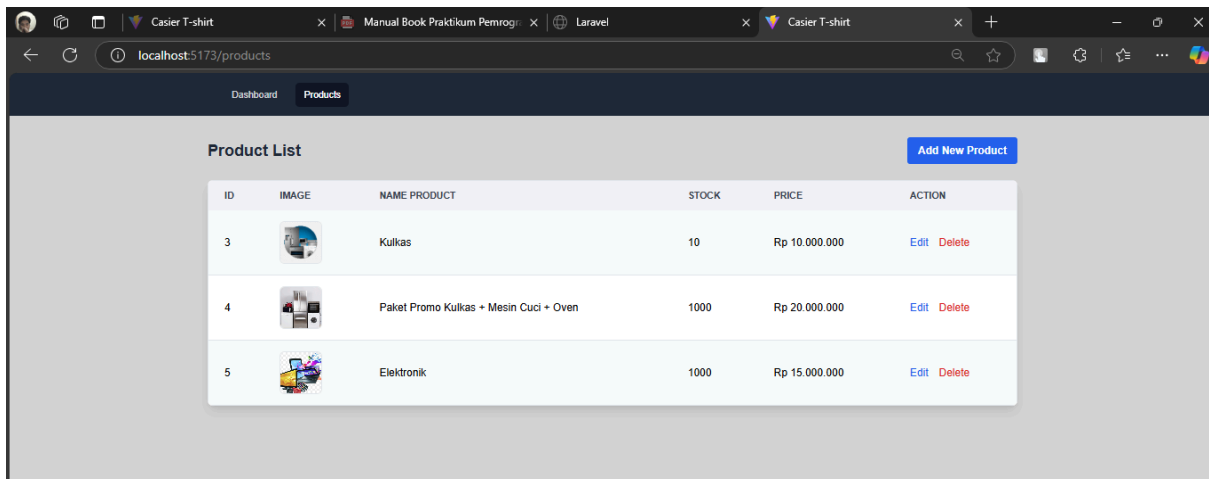
Bagaimana cara membuat proses delete data di dalam React menggunakan Rest API.

- Langkah 1 –Membuat Proses Delete Data

Disini kita akan menambahkan sebuah method baru di dalam halaman post index, dimana method tersebut nantinya akan dipanggil jika button delete diklik. Langsung saja, silahkan buka file src/views/posts/index.jsx. komponen React yang menampilkan daftar produk dari API dengan opsi untuk mengedit dan menghapus produk. Pengguna dapat melihat informasi produk seperti ID, gambar, nama, stok, dan harga dalam tabel, dan dapat mengedit atau menghapus produk tersebut. Jika produk berhasil dihapus, daftar produk akan diperbarui. Jika tidak ada produk yang tersedia, pesan "No products available!" akan ditampilkan. Terdapat juga tombol untuk menambahkan produk baru ke dalam daftar.



sesudah dihapus:



**Link Github :** [https://github.com/tarmidzibariq/ujian\\_akhir](https://github.com/tarmidzibariq/ujian_akhir)

Pembagian tugas :

1. Gilberto Patrick Lie = Front end dan manual book
2. Muhammad Tarmidzi Bariq = Backend dan manual book
3. Muhammad Farhan Zidan = Front end dan manual book

## **BAB IV**

### **PENUTUP**

#### **4.1 Kesimpulan**

Aplikasi Pemrograman Website Virtual Storage Menggunakan Laravel dan React dirancang untuk menyediakan layanan penyimpanan file berbasis cloud, di mana pengguna dapat mengunggah, menyimpan, mengedit, dan mengelola berbagai jenis file seperti gambar, dokumen, dan lainnya. Dengan Laravel sebagai kerangka kerja back-end, aplikasi ini menangani logika server-side, validasi data, otentikasi pengguna, serta interaksi dengan database. Sedangkan React digunakan untuk membangun antarmuka pengguna (UI) yang interaktif dan responsif, memungkinkan pengguna untuk dengan mudah mengakses, mengelola, dan memproses file secara visual melalui aplikasi web. Fitur utama termasuk pengelolaan produk, penyimpanan file, preview gambar, validasi input, serta kemampuan untuk menambahkan, mengedit, dan menghapus data secara dinamis.

#### **4.2 Saran**

Untuk memaksimalkan pembelajaran, disarankan agar pembaca memahami dasar-dasar React terlebih dahulu sebelum melanjutkan ke materi lanjutan seperti integrasi dengan Rest API. Praktikkan setiap langkah yang dibahas agar pemahaman lebih mendalam. Selain itu, eksplorasi dokumentasi resmi dari React, Axios, dan Laravel untuk mendapatkan informasi yang lebih lengkap.