|  |
| --- |
| **Kelas B** |

**LAPORAN PRAKTIKUM**

**SISTEM INFORMASI MANAJEMEN**

**Modul 7 & 8: Bootstrap & Github**



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Nama Praktikan** | **Nomor Mahasiswa** | **Tanggal Kumpul** | **Tanda Tangan**  **Praktikan** | |
| Ajeng Fatmah Azahra Muklis | 23611017 | 24/12/2024 |  | |
|  |  |  |  |  |
| **Nama Penilai** | **Tanggal Koreksi** | **Nilai** | **Tanda tangan** | |
| **Asisten** | **Dosen** |
| Lidya Palupi Saraswasti  Widya Saputri Agustin |  |  |  |  |
| Ghiffari Ahnaf Danarwindu, M.Sc. |  |  |  |  |

**JURUSAN STATISTIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS ISLAM INDONESIA**

**YOGYAKARTA**

**2024**

Daftar Isi

Daftar Isi i

Daftar Gambar ii

1 Pendahuluan 3

1.1 CSS 3 (Cascading Style Sheet 3) 3

1.2 Laravel 3

2 Deskripsi Kerja 6

2.1 Studi Kasus 6

2.2 Langkah Kerja 6

3 Pembahasan 24

3.1 Pembahasan Studi Kasus 24

3.1.1 Pembahasan Studi Kasus 24

3.1.2 Implementasi Form Input 25

3.1.3 Pengolahan Data di backend 26

3.1.4 Penampilan Data 28

3.2 Hasil Implementasi 28

4 Penutup 30

4.1 Kesimpulan 30

5 Daftar Pustaka 31

Daftar Gambar

[**Gambar 2.1** Logo XAMPP *Control Panel* 7](#_Toc184737962)

[**Gambar 2.2** Mengaktifkan *Apache* dan *MySQL* 7](#_Toc184737963)

[**Gambar 2.3** Tampilan Awal *phpMyAdmin* 8](#_Toc184737964)

[**Gambar 2.4.** Instalasi Composer 8](#_Toc184737965)

[**Gambar 2.5.** Memastikan composer sudah terinstall 9](#_Toc184737966)

[**Gambar 2.6.** Memastikan laravel sudah terinstall 9](#_Toc184737967)

[**Gambar 2.7.** Logo Visual Studio Code 9](#_Toc184737968)

[**Gambar 2.8** Path untuk membuat project 9](#_Toc184737969)

[**Gambar 2.9** Sintaks laravel untuk membuat project baru 10](#_Toc184737970)

[**Gambar 2.10** Opsi pada setup laravel 1 10](#_Toc184737971)

[**Gambar 2.11** Opsi pada setup laravel 2 10](#_Toc184737972)

[**Gambar 2.13** Opsi pada setup laravel 3 11](#_Toc184737973)

[**Gambar 2.14** Tampilan project setup laravel yang berhasil 11](#_Toc184737974)

[**Gambar 2.15** Menjalankan laravel 11](#_Toc184737975)

[**Gambar 2.16** Membuat database baru 12](#_Toc184737976)

[**Gambar 2.17** Menjalanan migrate database 12](#_Toc184737977)

[**Gambar 2.18** Query sql untuk membuat tabel 13](#_Toc184737978)

[**Gambar 2.19** Sintaks untuk Controller 14](#_Toc184737979)

[**Gambar 2.20** Sintaks untuk mengatur *route* 15](#_Toc184737980)

[**Gambar 2.21** Sintaks untuk mengatur model 16](#_Toc184737981)

[**Gambar 2.22** Sintaks untuk membuat formulir 17](#_Toc184737982)

[**Gambar 2.23** Sintaks untuk menampilkan hasil input formulir 18](#_Toc184737983)

[**Gambar 2.24** Sintaks CSS untuk visual form 23](#_Toc184737984)

[**Gambar 3.1** Struktur Tabel Products pada phpMyAdmin 25](#_Toc184737985)

[**Gambar 3.2** Tampilan halaman untuk menampilkan hasil input data 26](#_Toc184737986)

[**Gambar 3.3** Fungsi store : controller 27](#_Toc184737987)

[**Gambar 3.4** Tampilan Daftar Produk di Laravel 28](#_Toc184737988)

[**Gambar 3.5** Input data 29](#_Toc184737989)

[**Gambar 3.6** Notifikasi Data Berhasil Disimpan 29](#_Toc184737990)

[**Gambar 3.7** Data Berhasil Disimpan di Database 29](#_Toc184737991)

# **Pendahuluan**

## **Bootstrap**

Cascading Style Sheets (CSS) sangat penting dalam pengembangan web, menyediakan cara yang kuat untuk mengontrol tampilan elemen HTML. Sintaks CSS mencakup selektor, properti, dan nilai, yang sangat penting untuk mendefinisikan gaya dengan efektif. Selektor CSS memungkinkan pengembang untuk menargetkan elemen spesifik, seperti kelas dan ID, yang memungkinkan penataan yang tepat di seluruh halaman web (Rob, 2008). Berbagai properti CSS, termasuk warna dan tata letak, menentukan bagaimana elemen-elemen ini tampil, menawarkan berbagai pilihan kustomisasi. Selain itu, satuan CSS, seperti piksel dan persentase, sangat penting untuk menciptakan desain responsif yang dapat menyesuaikan dengan ukuran layar yang berbeda. Untuk fungsionalitas yang lebih baik, preprocessor CSS seperti Sass dan Less memungkinkan kode yang lebih modular dan efisien, dengan menambahkan fitur seperti variabel dan nested. Bersama-sama, komponen-komponen ini menjadikan CSS alat dasar untuk desain web modern.

## **Laravel**

Laravel adalah kerangka kerja PHP yang kuat dan elegan, dirancang untuk menyederhanakan pengembangan aplikasi web secara efisien. Dibangun dengan arsitektur MVC (Model-View-Controller), Laravel menyediakan serangkaian fitur yang memungkinkan pengembang membuat aplikasi web yang lengkap, skalabel, dan aman dengan mudah. Laravel dilengkapi alat seperti Artisan CLI untuk tugas-tugas berbasis command-line, sistem routing yang intuitif, middleware untuk memfilter permintaan, dan Eloquent ORM untuk manajemen basis data. Dengan dukungan untuk fitur-fitur seperti struktur modular, kemampuan pengujian, manajemen konfigurasi, dan query builder, Laravel membantu pengembang menulis kode yang bersih dan mudah dirawat. (Rahmat, n.d.)

Kerangka kerja ini terintegrasi dengan kebutuhan pengembangan web modern, menyediakan alat untuk menangani autentikasi pengguna, validasi data, manajemen sesi, dan pengunggahan file. Blade, mesin templating Laravel, mempermudah pembuatan tampilan dinamis, sementara dukungannya terhadap lokalisasi dan interaksi AJAX memungkinkan pengembang membangun antarmuka yang interaktif dan ramah pengguna. Laravel juga menawarkan mekanisme error handling dan logging yang andal, memastikan aplikasi dapat di-debug dan dikelola dengan efisien.

Pengembang dapat menggunakan alat migrasi dan seeding Laravel untuk mengelola skema basis data dengan efektif, sementara fitur bawaan untuk integrasi email dan event handling memberikan cara yang kuat untuk menerapkan fungsionalitas asinkron dan berbasis event. Laravel sangat memperhatikan keamanan, dengan fitur-fitur seperti perlindungan terhadap SQL injection, cross-site scripting (XSS), dan cross-site request forgery (CSRF).

Arsitektur modular dan struktur fleksibel Laravel mendorong skalabilitas, memungkinkan pengembang membangun aplikasi mulai dari proyek kecil hingga sistem perusahaan besar. Baik dalam pengaturan routing, membangun controller, atau bekerja dengan API RESTful, Laravel menawarkan solusi yang jelas, konsisten, dan terdokumentasi dengan baik untuk setiap aspek pengembangan aplikasi web. Dengan penekanan pada kesederhanaan, efisiensi, dan praktik terbaik, Laravel menjadi alat yang sangat berharga bagi pengembang yang ingin menciptakan aplikasi web modern berkinerja tinggi (Styo Promez, n.d.).

# **Deskripsi Kerja**

## **Studi Kasus**

Dalam studi kasus ini, Praktikan diminta mengembangkan sebuah aplikasi web sederhana yang terintegrasi dengan database untuk menampilkan konten dinamis. Aplikasi ini akan dibangun dengan dua komponen utama:

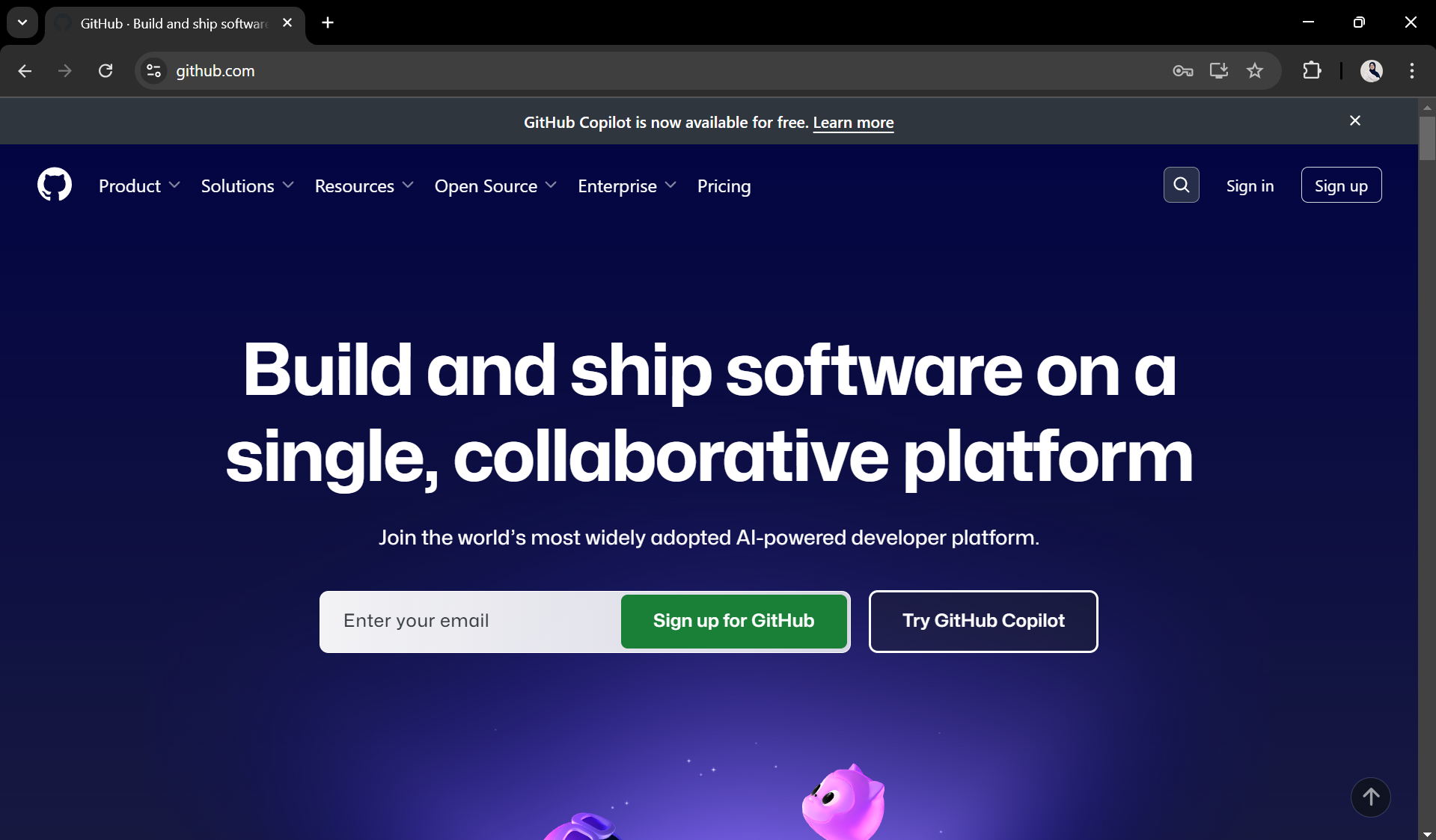
1. Frontend (HTML & CSS3): Mendesain antarmuka pengguna (UI) yang menarik dan responsif, memastikan tata letak dan elemen visual web dapat beradaptasi dengan ukuran layar perangkat.
2. Backend (Laravel): Mengelola data secara dinamis dari database menggunakan Laravel. Data akan disimpan dan diambil dari database, kemudian ditampilkan secara dinamis pada halaman web menggunakan Laravel.

Hasil akhirnya adalah aplikasi web yang memiliki desain menarik dengan tampilan data yang diambil dari database, yang dapat menyesuaikan tampilan berdasarkan ukuran layar

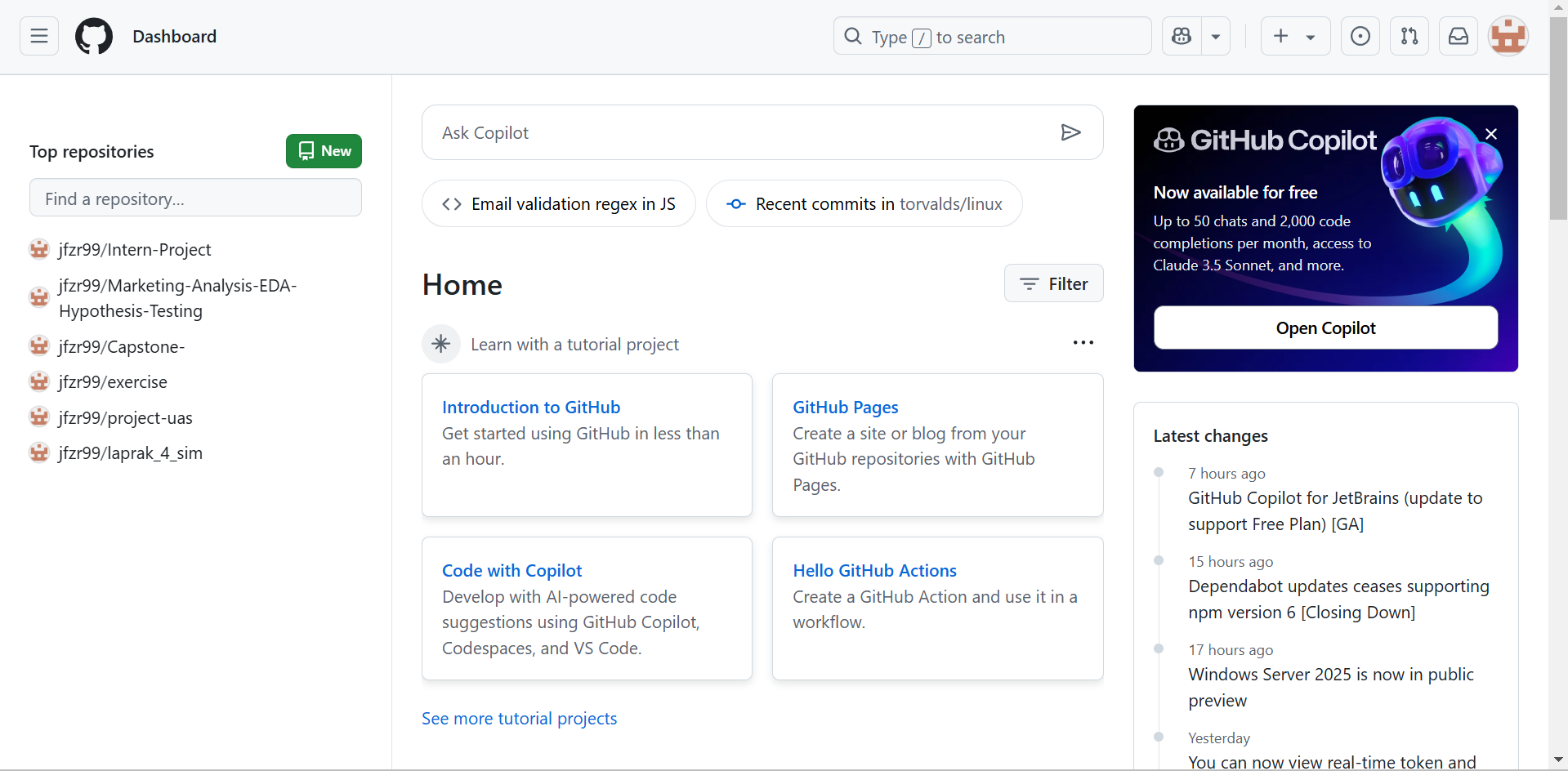
perangkat pengguna.

## **Langkah Kerjagi**

1. Pertama-tama, praktikan akan memulai dengan membuat website terlebih dahulu menggunakan html5 dan css+bootstrap,
2. Langkah pertama yang praktikan lakukan untuk menginiasi push file ke github adalah membuat akun github terlebih dahulu jika belum membuat, pergi ke halaman [www.github.com](http://www.github.com) dan lakukan pendaftaran:

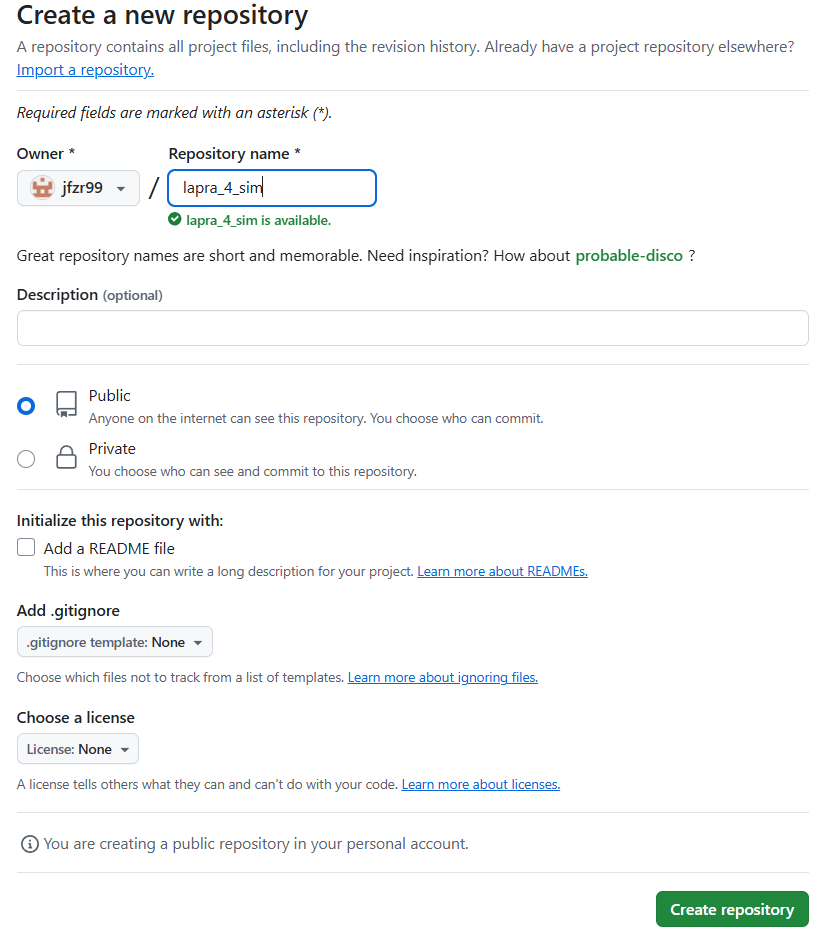


**Gambar 2.1** Halaman utama web github

1. Jika sudah mendaftar, berikut tampilan halaman awal praktikan

Selanjutnya praktikan akan membuat repository baru untuk membuat project ini yang praktikan namakan laprak\_4\_sim dengan menekan tombol new yang bewarna hijau di kiri atas

1. Lalu praktikan memasukkan beberapa informasi seperti nama repositori, selebihnya biarkan settingannya mengikuti default dari github untuk project yang sejenis ini, lalu praktikan menekan create repository untuk membuat projectnya.



1. Tahap selanjutnya ketika sudah mempunyai repository, praktikan akan menginisiasi project pada git bash di local computer yang akan di push ke repository github yang sudah dibuat, namun sebelum itu, praktikan aka

# **Pembahasan**

## **Pembahasan Studi Kasus**

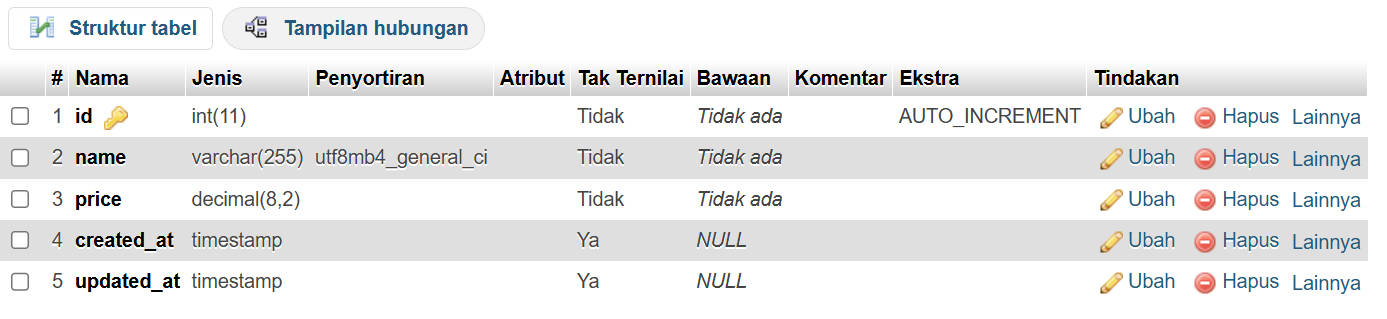
Dalam studi kasus ini, praktikan diminta untuk mengimplementasikan sistem berbasis Laravel yang mencakup proses pengisian data melalui form, penyimpanan ke dalam database MySQL, dan penampilan data melalui antarmuka web. Studi kasus melibatkan beberapa tahapan, antara lain: pembuatan tabel database, implementasi form dengan validasi, pengolahan data di backend, dan penampilan data menggunakan **blade template** di Laravel.

### **Pembahasan Studi Kasus**

Langkah pertama dalam studi kasus ini adalah membuat database bernama **"testproject"** pada MySQL, dengan sebuah tabel bernama **"products"** yang dirancang untuk menyimpan data produk. Tabel ini memiliki struktur sebagai berikut:

* **id**: Kolom utama bertipe INT dengan atribut PRIMARY KEY dan AUTO\_INCREMENT. Kolom ini digunakan untuk memberikan identitas unik pada setiap entri data.
* **name**: Kolom bertipe VARCHAR dengan panjang maksimal 255 karakter untuk menyimpan nama produk.
* **price**: Kolom bertipe INT untuk menyimpan harga produk dalam bentuk angka.
* **user\_id**: Kolom bertipe UNSIGNED BIGINT yang bertindak sebagai FOREIGN KEY, menghubungkan data produk dengan pengguna tertentu pada tabel **users**.

Tujuan desain tabel ini adalah untuk menyimpan data dengan struktur yang sederhana, tetapi tetap mendukung fitur multi-pengguna. Atribut FOREIGN KEY memastikan bahwa setiap data produk terhubung dengan pengguna spesifik, sehingga mendukung keamanan dan personalisasi data. Tabel ini dibuat menggunakan perintah SQL sebagai berikut:



**Gambar 3.1** Struktur Tabel Products pada phpMyAdmin

Pembuatan tabel dilakukan menggunakan perintah SQL pada **Gambar 2.18** di langkah kerja. Pembuatan tabel ini penting untuk memastikan bahwa data yang diterima dari pengguna dapat disimpan secara terstruktur dan siap untuk digunakan pada langkah-langkah selanjutnya.

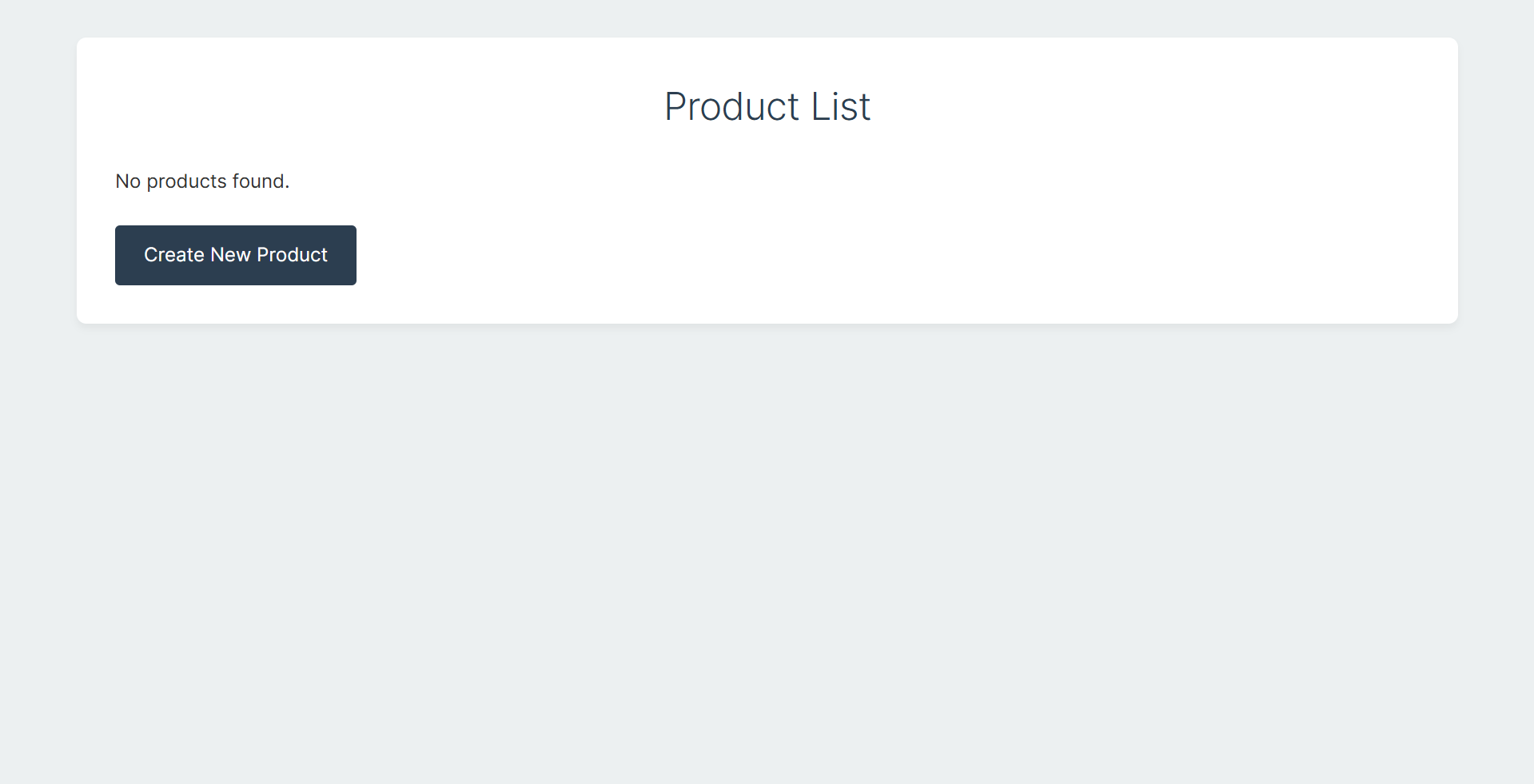
### **Implementasi Form Input**

Bagian kedua dari studi kasus ini adalah pembuatan antarmuka form input menggunakan file **create.blade.php**. Form ini dirancang menggunakan kombinasi HTML dan CSS untuk memberikan pengalaman pengguna yang responsif dan mudah digunakan. Form berisi kolom input berikut:

* **Nama Produk**: Menggunakan elemen <input type="text"> untuk menerima nama produk.
* **Harga Produk**: Menggunakan elemen <input type="number"> untuk menerima harga produk dalam format angka.

Setiap input pada form dilengkapi atribut required untuk memastikan pengguna mengisi semua data yang diperlukan sebelum data dikirim ke server. Data dikirimkan melalui metode POST ke endpoint yang ditentukan dalam route Laravel.

Form ini didesain agar ramah pengguna dengan tata letak yang rapi dan elemen yang mudah diakses. Sebagai contoh, tombol kirim data dirancang menggunakan CSS untuk memastikan tampilannya menarik dan memberikan umpan balik visual ketika diklik.



**Gambar 3.2** Tampilan halaman untuk menampilkan hasil input data

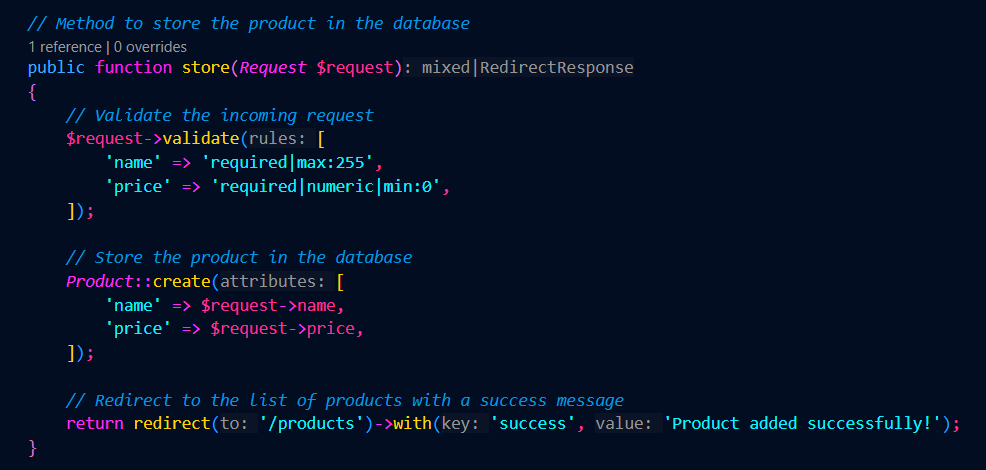
### **Pengolahan Data di backend**

Langkah berikutnya adalah pengolahan data yang diterima dari form pada backend Laravel. Pengolahan dilakukan oleh controller **ProductController**, khususnya dalam fungsi **store**. Berikut langkah-langkah yang dilakukan pada backend:

1. **Validasi Data**: Data yang diterima dari form divalidasi menggunakan metode Laravel validate() untuk memastikan bahwa data sesuai dengan aturan yang ditentukan, seperti panjang maksimal dan tipe data.
2. **Penyimpanan Data**: Setelah validasi berhasil, data disimpan ke dalam tabel **products** di database menggunakan model **Product**. Data yang disimpan mencakup nama produk, harga, dan user\_id pengguna yang sedang login.
3. **Redirect dan Feedback**: Setelah data berhasil disimpan, pengguna diarahkan kembali ke halaman daftar produk dengan pesan sukses menggunakan metode **redirect()**.

Pengolahan data pada backend dilakukan melalui ProductController, yang berfungsi sebagai penghubung antara form dan database. Praktikan menambahkan fungsi store pada controller untuk menangani logika penyimpanan data. Fungsi ini memvalidasi data yang dikirim oleh form dengan aturan tertentu, seperti memastikan kolom name memiliki panjang maksimum 255 karakter dan kolom price hanya menerima nilai numerik.

Jika data valid, fungsi ini akan menyimpan data ke tabel products dengan menambahkan ID pengguna yang sedang login. Hal ini memungkinkan data setiap pengguna dikelola secara individual. Implementasi fungsi store memastikan proses input data berjalan lancar dengan keamanan yang terjaga.

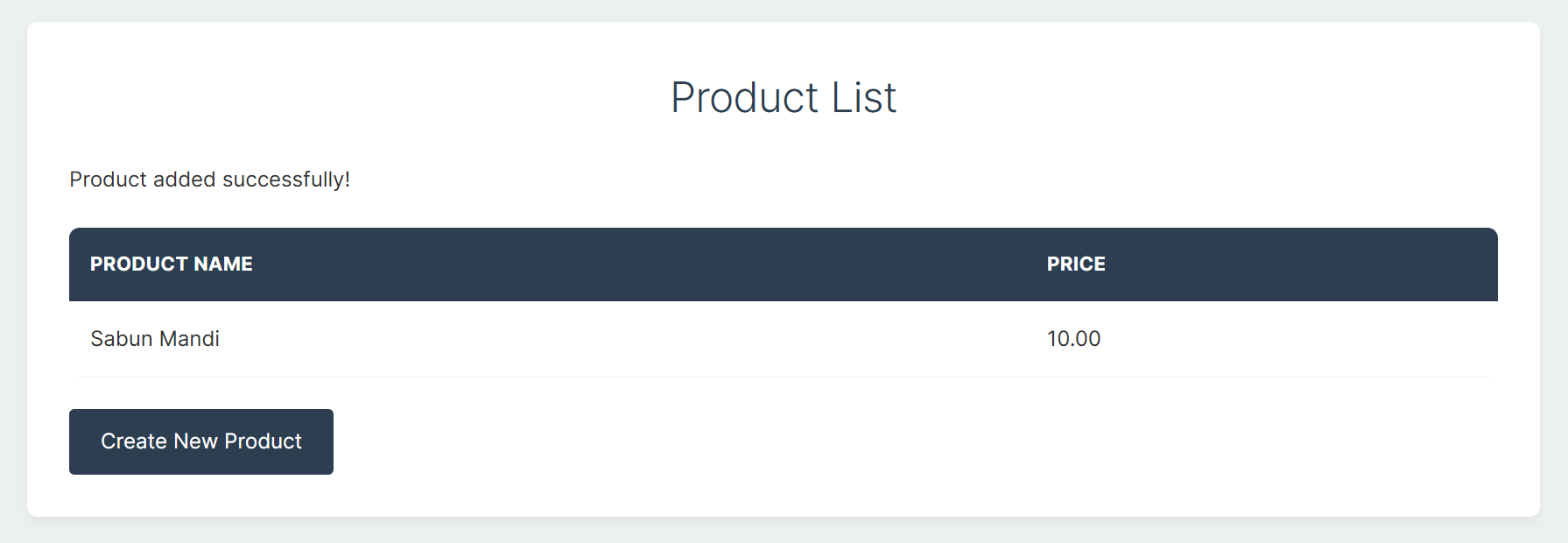


**Gambar 3.3** Fungsi store : controller

### **Penampilan Data**

Untuk menampilkan data yang telah disimpan, praktikan membuat file index.blade.php. File ini digunakan untuk mempresentasikan data secara dinamis dalam bentuk tabel yang rapi. Praktikan memastikan bahwa data yang ditampilkan hanya milik pengguna yang sedang login, sehingga privasi data tetap terjaga.

Proses pengambilan data dilakukan menggunakan Eloquent ORM milik Laravel, yang menyediakan metode sederhana untuk menyaring data berdasarkan ID pengguna. Tampilan tabel pada halaman ini dirancang agar memudahkan pengguna dalam membaca dan mengelola data yang telah mereka input.

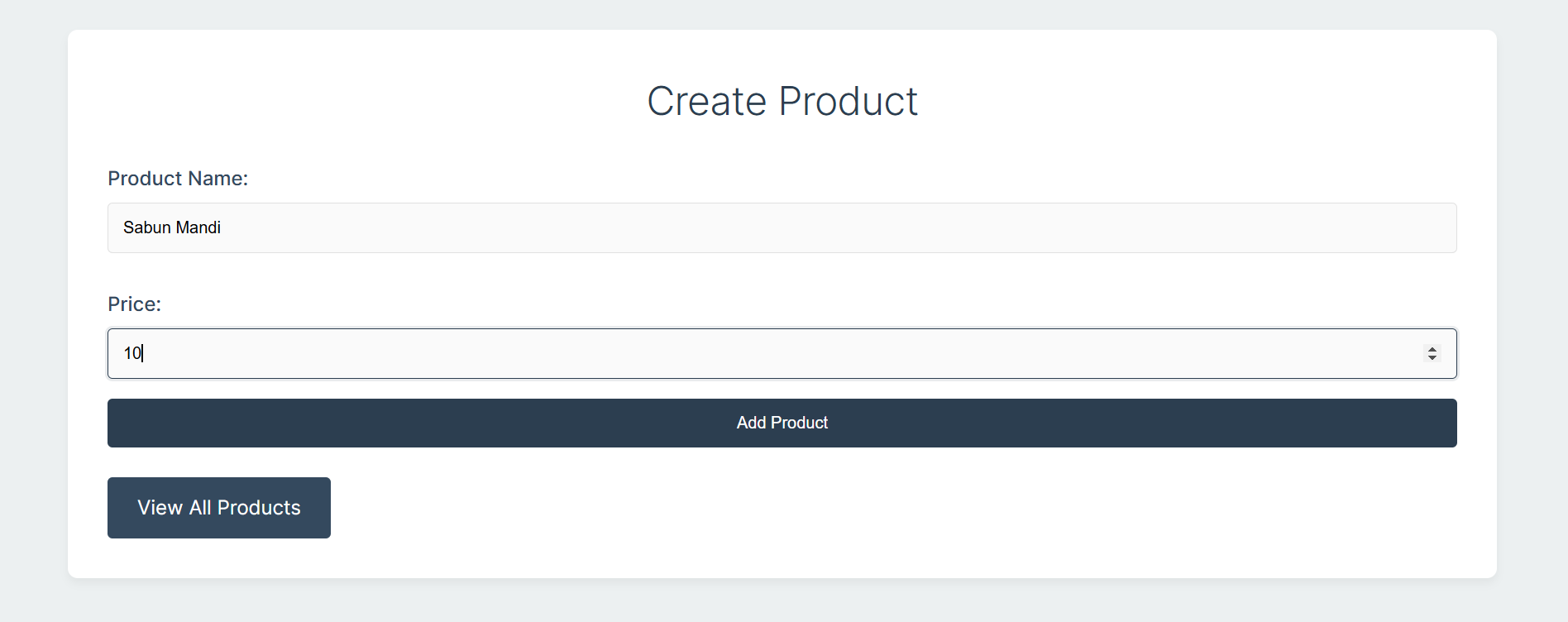


**Gambar 3.4** Tampilan Daftar Produk di Laravel

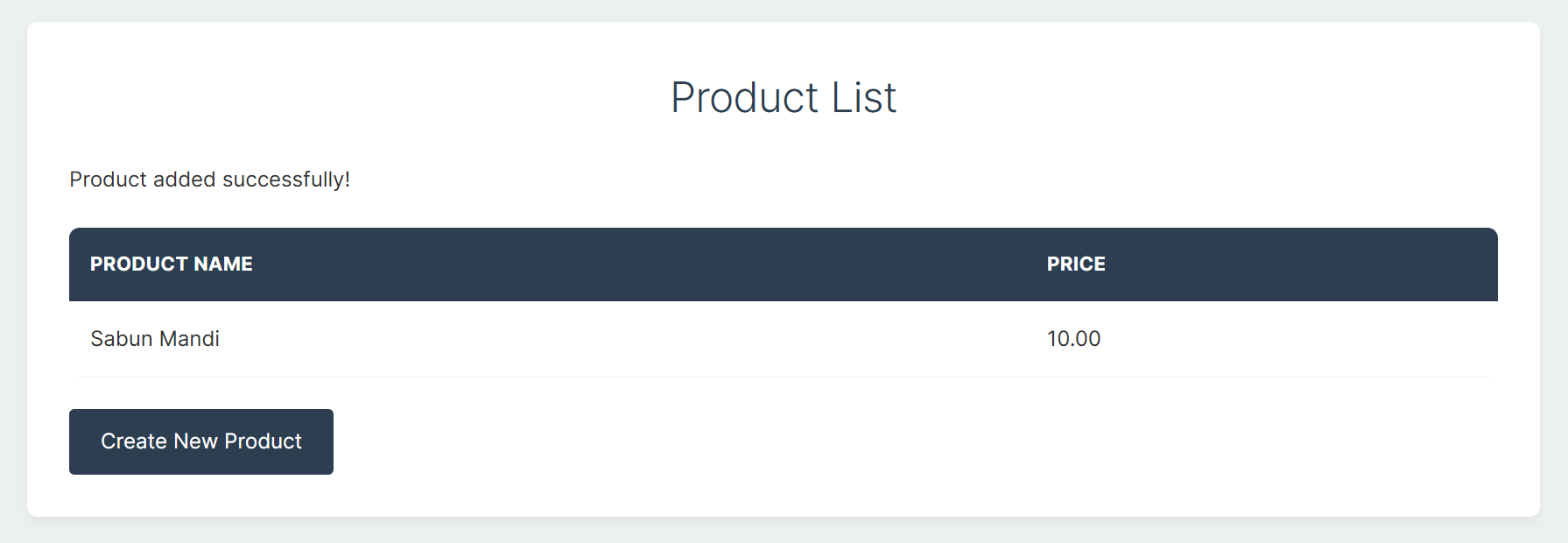
## **Hasil Implementasi**

Hasil implementasi menunjukkan bahwa sistem Laravel berhasil memenuhi seluruh fungsi yang dirancang. Berikut ini adalah pencapaian utama yang diperoleh:

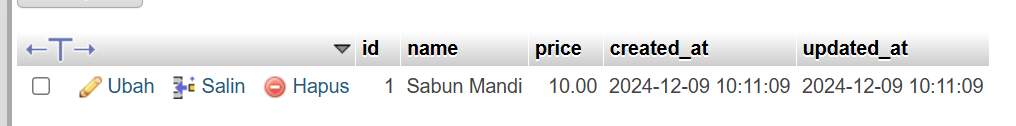
1. **Input Data:** Data produk dapat dimasukkan melalui form input dengan validasi yang sesuai.
2. **Penyimpanan Data:** Data yang valid disimpan ke dalam tabel **products**, lengkap dengan ID pengguna untuk personalisasi.
3. **Penampilan Data:** Data produk yang relevan ditampilkan secara dinamis dalam tabel pada halaman **index.blade.php**, hanya mencakup data milik pengguna yang sedang login.



**Gambar 3.5** Input data



**Gambar 3.6** Notifikasi Data Berhasil Disimpan



**Gambar 3.7** Data Berhasil Disimpan di Database

# **Penutup**

## **Kesimpulan**

1. Praktikan berhasil mengimplementasikan sistem manajemen produk menggunakan Laravel dan MySQL, dengan memanfaatkan komponen **controller**, **model**, **routes**, dan **views** yang terintegrasi secara efektif.
2. **Controller** berfungsi mengelola logika aplikasi, **model** menghubungkan data dengan database melalui Eloquent ORM, **routes** mengatur akses fitur, dan **views** menyediakan antarmuka pengguna yang dinamis.
3. Sistem berhasil memungkinkan pengguna menginput, menyimpan, dan menampilkan data produk secara terstruktur dan aman. Penggunaan fitur autentikasi memastikan hanya data relevan yang dapat diakses oleh pengguna tertentu.
4. Secara keseluruhan, Laravel terbukti andal dalam mengembangkan aplikasi berbasis web dengan arsitektur MVC yang efisien dan keamanan data yang terjamin.

# **Daftar Pustaka**

*(17) Basics of Laravel Framework - Practical Guides | styo promez - Academia.edu*. (n.d.). Retrieved December 8, 2024, from https://www.academia.edu/38613913/Basics\_of\_Laravel\_Framework\_Practical\_Guides

Rahmat, F. (n.d.). *Laravel*. Retrieved December 9, 2024, from https://rahmatfauzi.staff.telkomuniversity.ac.id/files/2018/11/2.-Rahmat-Fauzi-Arsitektur-LARAVEL.pdf

Rob, H. (2008). *HTML, XHTML, and CSS: Your Visual Blueprint for Designing Effective Web Pages.*