LAPORAN PRAKTIKUM

MATA KULIAH PEMROGRAMAN WEB LANJUT

Dosen Pengampu : Dimas Wahyu Wibowo, S.T., M.T.

JOBSHEET I : MEMBUAT CRUD LARAVEL SEDERHANA DENGAN ELOQUENT ORM

Disusun oleh :

Nama : Vita Eka Saraswati

NIM : 2341760082

No Absen : 29

JURUSAN TEKNOLOGI INFORMASI

PROGRAM STUDI D-IV SISTEM INFORMASI BISNIS

POLITEKNIK NEGERI MALANG

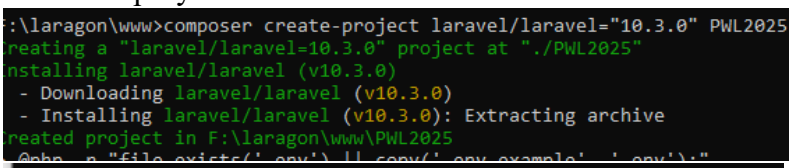
2025

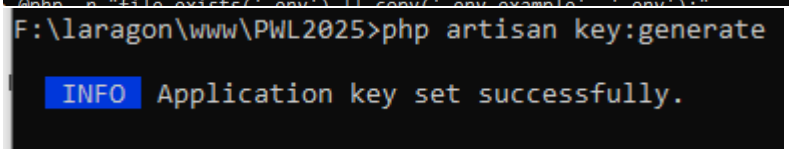
I. INSTALASI

1. Instalasi Composer dan Membuat Project Framework Laravel

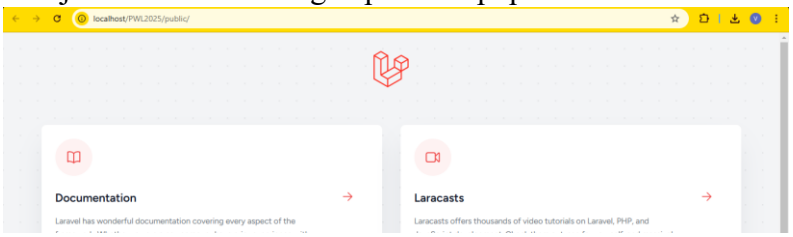
II. PERSIAPAN PROYEK

1. Membuat proyek Laravel



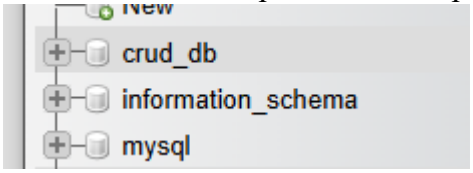


2. Menjalankan server dengan perintah php artisan serve



III. MEMBUAT DATABASE

1. Membuat Database pada localhost/phpmyadmin



2. Konfigurasi Database

```
DB_CONNECTION=mysql
DB_HOST=localhost
DB_PORT=3306
DB_DATABASE=crud_db
DB_USERNAME=root
DB_PASSWORD=
```

IV. MEMBUAT MODEL DAN MIGRASI

1. Membuat Model dan Migrasi

```
PS F:\laragon\www\PWL2025> php artisan make:model Item -m

  INFO  Model [F:\laragon\www\PWL2025\app\Models\Item.php] created successfully.

  INFO  Migration [F:\laragon\www\PWL2025\database\migrations/2025_03_04_132340_create_items_table.php] created successfully.
```

2. Mengedit File Migrasi

```php
public function up(): void //definisi fungsi up saat migration dijalankan
{
    Schema::create(table: 'items', callback: function (Blueprint $table): void { //Membuat tabel baru dengan nama items
        $table->id(); //membuat kolom id sbg primary key
        $table->string(column: 'name'); // membuat kolom name dengan tipe data VARCHAR string
        $table->text(column: 'description'); //membuat kolom description dengan tipe data TEXT
        $table->timestamps(); //menambahkan dua kolom otomatis: created_at dan updated_at untuk mencatat waktu pembuatan dan pembaruan data.

    });
}
```

3. Menjalankan Migrasi

```
PS F:\laragon\www\PWL2025> php artisan migrate

  INFO  Preparing database.

  Creating migration table ..................................................... 228ms DONE

  INFO  Running migrations.

  2014_10_12_000000_create_users_table ........................................ 237ms DONE
  2014_10_12_100000_create_password_reset_tokens_table ......................... 36ms DONE
  2019_08_19_000000_create_failed_jobs_table ................................... 426ms DONE
  2019_12_14_000001_create_personal_access_tokens_table ....................... 327ms DONE
  2025_03_04_132340_create_items_table ........................................ 139ms DONE
```

V. MEMBUAT CONTROLLER

1. Membuat Controller

```
PS F:\laragon\www\PWL2025> php artisan make:controller ItemController --resource

  INFO  Controller [F:\laragon\www\PWL2025\app\Http\Controllers\ItemController.php] created successfully.
```

2. Implementasi CRUD

app > Http > Controllers > ItemController.php > PHP Intelephense > ItemController

```php
<?php

namespace App\Http\Controllers; //Menetapkan namespace App\Http\Controllers.

use App\Models\Item; ////menggunakan model Item dari Laravel
use Illuminate\Http\Request; //menggunakan model request dari Laravel

2 references | 0 implementations
class ItemController extends Controller //deklarasi ItemController yang mewarisi Controller
{

    0 references | 0 overrides
    public function index(): Factory|View //deklarasi fungsi index
    {
        $items = Item::all(); //impor semua data Item
        return view(view: 'items.index', data: compact(var_name: 'items')); //impor semua data Item ke view items.index
    }

    public function create(): Factory|View //deklarasi fungsi membuat item baru
    {
        return view(view: 'items.create'); //impor data yang dibuat ke view items.create
    }

    0 references | 0 overrides
    public function store(Request $request): RedirectResponse //method store
    {
        $request->validate(rules: [ //validasi input (name dan description wajib diisi).
            'name' => 'required',   //nama wajib diisi
            'description' => 'required', //deskripsi wajib diisi
        ]);


        // Hanya masukkan atribut yang diizinkan
        Item::create(attributes: $request->only(keys: ['name', 'description']));
        return redirect()->route(route: 'items.index')->with(key: 'success', value: 'Item added successfully.'); //Redirect ke halaman index, dan displa
    }
```

```php
    0 references | 0 overrides
    public function show(string $id): Factory|View // Method untuk menampilkan halaman detail item
    {
        return view(view: 'items.show', data: compact(var_name: 'item')); // Mengirim data item ke tampilan 'items.show'
    }

    0 references | 0 overrides
    public function edit(string $id): Factory|View //// Method untuk menampilkan halaman edit item
    {
        return view(view: 'items.edit', data: compact(var_name: 'item')); // Mengirim data item ke tampilan 'items.edit'
    }

    0 references | 0 overrides
    public function update(Request $request, Item $item): RedirectResponse// Method untuk memperbarui item
    {
        $request->validate(rules: [
            'name' => 'required', // Validasi: name wajib diisi
            'description' => 'required', // Validasi: description wajib diisi
        ]);

        // Hanya masukkan atribut yang diizinkan
        $item->update(attributes: $request->only(keys: ['name', 'description']));  // Update item dengan data yang valid
        return redirect()->route(route: 'items.index')->with(key: 'success', value: 'Item updated successfully.'); // Update item dengan data yang valid,
    }

    0 references | 0 overrides
    public function destroy(Item $item): RedirectResponse //// Method untuk hapus item
    {
        // return redirect()->route('items.index');
        $item->delete(); // Hapus item dari database
        return redirect()->route(route: 'items.index')->with(key: 'success', value: 'Item deleted successfully.'); // Redirect ke halaman daftar item, disp
    }
}
```

VI. MEMBUAT ROUTING

1. Mengedit File Routing

routes > web.php > ...

```php
<?php

use Illuminate\Support\Facades\Route;  // Mengimpor kelas Route untuk mendefinisikan rute
use App\Http\Controllers\ItemController; // Mengimpor ItemController untuk digunakan dalam rute

Route::get(uri: '/', action: function (): Factory|View { // Mendefinisikan rute untuk halaman utama "/"
    return view(view: 'welcome'); // Menampilkan tampilan 'welcome'
});

Route::resource(name: 'items', controller: ItemController::class); // Membuat rute resource untuk ItemController
```

2. Membuat View dengan membuat folder baru bernama items berisikan 4 file, antara lain:
   a. Index blade.php

resources > views > items > index.blade.php

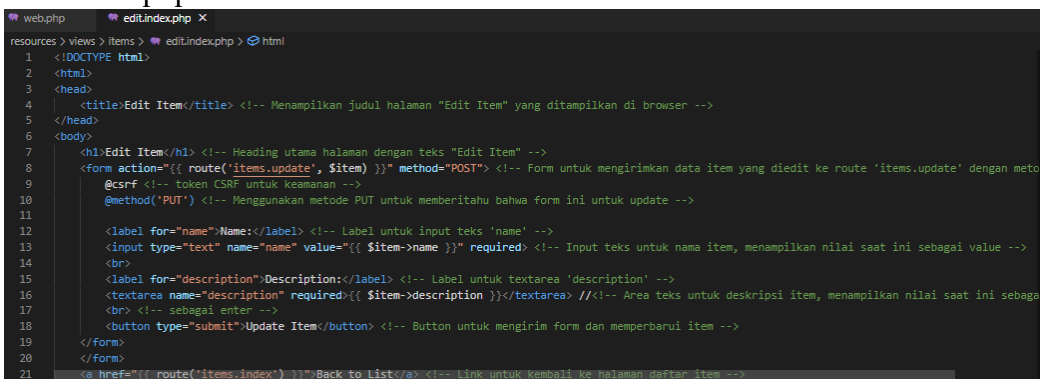
```php
<!DOCTYPE html>
<html>
<head>
    <title>Item List</title> <!-- Judul halaman yang ditampilkan di tab browser -->
</head>
<body>
    <h1>Items</h1> <!-- heading1 halaman dengan teks "Items" -->
    @if(session(key: 'success'))<!-- Mengecek apakah ada pesan sukses dalam session -->
        <p>{{ session(key: 'success') }}</p> <!-- Menampilkan pesan sukses jika ada -->
    @endif
    <a href="{{ route(name: 'items.create') }}">Add Item</a> <!-- Link untuk membuat item baru (mengarah ke halaman create) -->

    <ul>
        @foreach ($items as $item) <!-- Loop untuk menampilkan semua item dalam variabel $items -->
            <li>
                {{ $item->name }} - <!-- Menampilkan nama item -->
                <a href="{{ route(name: 'items.edit', parameters: $item) }}">Edit</a> <!-- Link untuk mengedit item, mengarah ke halaman edit -->
                <form action="{{ route(name: 'items.destroy', parameters: $item) }}" method="POST" style="display:inline;"> <!-- Form untuk menghapus item
                    @csrf <!-- Token CSRF-->
```

   b. Create blade.php

resources > views > items > create.blade.php

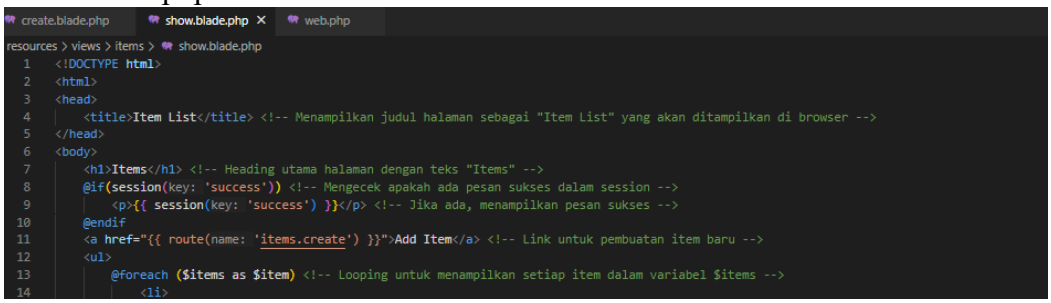
```php
<!DOCTYPE html>
<html>
<head>
    <title>Add Item</title> <!-- Menampilkan judul halaman "Add Item" yang ditampilkan di browser -->
</head>
<body>
    <h1>Add Item</h1> <!-- Heading utama halaman dengan teks "Add Item" -->
    <form action="{{ route(name: 'items.store') }}" method="POST"> <!-- Form untuk mengirim data item baru ke route 'items.store' dengan metode POST -->
        @csrf <!-- Menyertakan token CSRF untuk keamanan -->
        @csrf <!-- Menyertakan token CSRF untuk keamanan -->
        <label for="name">Name:</label> <!-- Label untuk input 'name' -->
        <input type="text" name="name" required> <!-- Input teks untuk nama item, wajib diisi -->
        <br> <!-- sebagai enter -->
        <label for="description">Description:</label> <!-- Label untuk textarea 'description' -->
        <textarea name="description" required></textarea> <!-- Area teks untuk deskripsi item, wajib diisi -->
        <br> <!-- sebagai enter -->
        <button type="submit">Add Item</button>  <!-- Button untuk mengirim form dan menambahkan item -->
    </form>
    <a href="{{ route(name: 'items.index') }}">Back to List</a> <!-- Link untuk kembali ke halaman daftar item -->
</body>
</html>
```

c. Edit blade.php



```
web.php  edit.index.php ×
resources > views > items > edit.index.php > html
1   <!DOCTYPE html>
2   <html>
3   <head>
4       <title>Edit Item</title> <!-- Menampilkan judul halaman "Edit Item" yang ditampilkan di browser -->
5   </head>
6   <body>
7       <h1>Edit Item</h1> <!-- Heading utama halaman dengan teks "Edit Item" -->
8       <form action="{{ route('items.update', $item) }}" method="POST"> <!-- Form untuk mengirimkan data item yang diedit ke route 'items.update' dengan meto
9           @csrf <!-- token CSRF untuk keamanan -->
10          @method('PUT') <!-- Menggunakan metode PUT untuk memberitahu bahwa form ini untuk update -->
11
12          <label for="name">Name:</label> <!-- Label untuk input teks 'name' -->
13          <input type="text" name="name" value="{{ $item->name }}" required> <!-- Input teks untuk nama item, menampilkan nilai saat ini sebagai value -->
14          <br>
15          <label for="description">Description:</label> <!-- Label untuk textarea 'description' -->
16          <textarea name="description" required>{{ $item->description }}</textarea> //<!-- Area teks untuk deskripsi item, menampilkan nilai saat ini sebaga
17          <br> <!-- sebagai enter -->
18          <button type="submit">Update Item</button> <!-- Button untuk mengirim form dan memperbarui item -->
19      </form>
20      </form>
21      <a href="{{ route('items.index') }}">Back to List</a> <!-- Link untuk kembali ke halaman daftar item -->
```

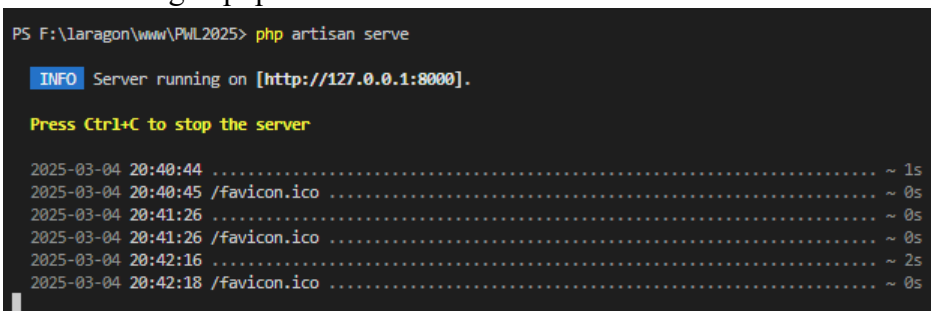
d. Show blade.php



```
create.blade.php  show.blade.php ×  web.php
resources > views > items > show.blade.php
1   <!DOCTYPE html>
2   <html>
3   <head>
4       <title>Item List</title> <!-- Menampilkan judul halaman sebagai "Item List" yang akan ditampilkan di browser -->
5   </head>
6   <body>
7       <h1>Items</h1> <!-- Heading utama halaman dengan teks "Items" -->
8       @if(session(key: 'success')) <!-- Mengecek apakah ada pesan sukses dalam session -->
9           <p>{{ session(key: 'success') }}</p> <!-- Jika ada, menampilkan pesan sukses -->
10      @endif
11      <a href="{{ route(name: 'items.create') }}">Add Item</a> <!-- Link untuk pembuatan item baru -->
12      <ul>
13          @foreach ($items as $item) <!-- Looping untuk menampilkan setiap item dalam variabel $items -->
14              <li>
```

VII. MENJALANKAN SERVER

1. Jalankan dengan php artisan serve



```
PS F:\laragon\www\PWL2025> php artisan serve

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

  Press Ctrl+C to stop the server

  2025-03-04 20:40:44 ..... ~ 1s
  2025-03-04 20:40:45 /favicon.ico ..... ~ 0s
  2025-03-04 20:41:26 ..... ~ 0s
  2025-03-04 20:41:26 /favicon.ico ..... ~ 0s
  2025-03-04 20:42:16 ..... ~ 2s
  2025-03-04 20:42:18 /favicon.ico ..... ~ 0s
```

2. Tampilan pada browser

Items

Add Item

Add Item

Name:

Description:

Add Item

Back to List