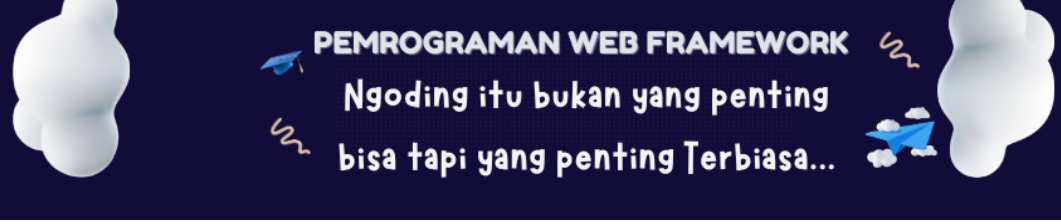
PENGEMBANGAN DAN PENYELENGGARAAN
PEMBELAJARAN DIGITAL (P3D)

Modul Pembelajaran

# SISTEM INFORMASI

## Pemrgoraman Web Framework


Disusun oleh : Wicaksono Yuli Sulistyo




# Modul Ajar: CRUD Dasar pada Laravel

## Pengenalan CRUD

CRUD adalah akronim yang merujuk pada empat operasi dasar yang sering dilakukan dalam aplikasi berbasis database: Create (Membuat), Read (Membaca), Update (Memperbarui), dan Delete (Menghapus). Operasi ini merupakan fondasi bagi banyak aplikasi web, dan Laravel, sebagai framework PHP yang powerful, menyediakan alat dan fitur untuk mempermudah implementasi CRUD. Dalam modul ini, kita akan membahas langkah-langkah untuk membuat aplikasi CRUD dasar dengan Laravel, mulai dari pengaturan awal hingga implementasi fungsionalitas lengkap.

## Persiapan Awal

Sebelum memulai, pastikan Anda telah melakukan beberapa langkah awal berikut:

1. **Instalasi Laravel**: Jika belum, buat proyek Laravel baru dengan perintah:

   ```
   bash

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

2. **Koneksi Database**: Konfigurasikan koneksi database di file `.env`. Misalnya, jika menggunakan MySQL, tambahkan detail berikut:

   ```
   plaintext

   DB_CONNECTION=mysql
   DB_HOST=127.0.0.1
   DB_PORT=3306
   DB_DATABASE=nama_database
   DB_USERNAME=username_database
   DB_PASSWORD=password_database
   ```

3. **Migrasi Database**: Jalankan migrasi untuk membuat tabel `users` yang sudah ada:

   ```
   bash

   php artisan migrate
   ```

## Membuat Migrasi untuk Tabel CRUD

Langkah pertama dalam membangun aplikasi CRUD adalah membuat migrasi untuk tabel yang akan kita gunakan. Dalam contoh ini, kita akan membuat tabel `posts` untuk menyimpan data artikel. Gunakan perintah berikut untuk membuat migrasi:

```
bash
```

```
php artisan make:migration create_posts_table
```

Setelah itu, buka file migrasi yang baru saja dibuat di `database/migrations`, dan tambahkan kolom yang diperlukan, seperti judul dan konten:

```
php

use Illuminate\Database\Migrations\Migration;
use Illuminate\Database\Schema\Blueprint;
use Illuminate\Support\Facades\Schema;

class CreatePostsTable extends Migration
{
    public function up()
    {
        Schema::create('posts', function (Blueprint $table) {
            $table->id();
            $table->string('title');
            $table->text('content');
            $table->timestamps();
        });
    }

    public function down()
    {
        Schema::dropIfExists('posts');
    }
}
```

Setelah mendefinisikan migrasi, jalankan perintah berikut untuk membuat tabel `posts` di database:

```
bash

php artisan migrate
```

## Membuat Model Eloquent

Setelah tabel dibuat, langkah selanjutnya adalah membuat model Eloquent untuk tabel `posts`. Model ini akan memudahkan kita dalam melakukan operasi CRUD. Gunakan perintah berikut untuk membuat model:

```
bash

php artisan make:model Post
```

Model `Post` akan berada di `app/Models/Post.php`. Model ini secara otomatis terhubung dengan tabel `posts` di database dan dapat digunakan untuk melakukan operasi data.

## Membuat Controller

Untuk mengelola logika aplikasi, kita perlu membuat controller. Controller ini akan menangani semua permintaan terkait CRUD. Gunakan perintah berikut untuk membuat controller:

```
bash

php artisan make:controller PostController
```

Controller `PostController` akan berada di `app/Http/Controllers/PostController.php`. Dalam controller ini, kita akan mendefinisikan metode untuk setiap operasi CRUD: `index`, `create`, `store`, `edit`, `update`, dan `destroy`.

Berikut adalah contoh implementasi metode di `PostController`:

```
php

namespace App\Http\Controllers;

use App\Models\Post;
use Illuminate\Http\Request;

class PostController extends Controller
{
    // Menampilkan semua post
    public function index()
    {
        $posts = Post::all();
        return view('posts.index', compact('posts'));
    }

    // Menampilkan form untuk membuat post baru
    public function create()
    {
        return view('posts.create');
    }

    // Menyimpan post baru ke database
    public function store(Request $request)
    {
        $request->validate([
            'title' => 'required',
            'content' => 'required',
        ]);

        Post::create($request->all());

        return redirect()->route('posts.index')
                         ->with('success', 'Post created successfully.');
    }

    // Menampilkan form untuk mengedit post
    public function edit(Post $post)
    {
        return view('posts.edit', compact('post'));
```

```
    }

    // Memperbarui post yang ada
    public function update(Request $request, Post $post)
    {
        $request->validate([
            'title' => 'required',
            'content' => 'required',
        ]);

        $post->update($request->all());

        return redirect()->route('posts.index')
                         ->with('success', 'Post updated successfully.');
    }

    // Menghapus post
    public function destroy(Post $post)
    {
        $post->delete();
        return redirect()->route('posts.index')
                         ->with('success', 'Post deleted successfully.');
    }
}
```

## Mengatur Rute

Langkah selanjutnya adalah mendefinisikan rute untuk aplikasi CRUD di file `routes/web.php`. Kita perlu membuat rute untuk setiap operasi CRUD yang telah kita buat di `PostController`. Contoh rute yang dapat ditambahkan adalah sebagai berikut:

```
php

use App\Http\Controllers\PostController;

Route::resource('posts', PostController::class);
```

Rute ini secara otomatis akan membuat semua rute yang diperlukan untuk operasi CRUD menggunakan konvensi RESTful.

## Membuat Tampilan

Setelah mendefinisikan rute dan controller, langkah berikutnya adalah membuat tampilan untuk setiap bagian aplikasi. Kita akan membuat tampilan untuk menampilkan semua post, formulir untuk membuat post baru, dan formulir untuk mengedit post.

1. **Tampilan Index (menampilkan semua post)**: Buat file `resources/views/posts/index.blade.php` dengan konten berikut:

   ```
   blade
   ```

```
@extends('layouts.app')

@section('content')
    <div class="container">
        <h1>Posts</h1>
        <a href="{{ route('posts.create') }}" class="btn btn-
primary">Create New Post</a>
        @if ($message = Session::get('success'))
            <div class="alert alert-success">{{ $message }}</div>
        @endif
        <table class="table">
            <thead>
                <tr>
                    <th>Title</th>
                    <th>Content</th>
                    <th>Action</th>
                </tr>
            </thead>
            <tbody>
                @foreach ($posts as $post)
                    <tr>
                        <td>{{ $post->title }}</td>
                        <td>{{ $post->content }}</td>
                        <td>
                            <a href="{{ route('posts.edit', $post->id)
}}" class="btn btn-warning">Edit</a>
                            <form action="{{ route('posts.destroy',
$post->id) }}" method="POST" style="display:inline;">
                                @csrf
                                @method('DELETE')
                                <button type="submit" class="btn btn-
danger">Delete</button>
                            </form>
                        </td>
                    </tr>
                @endforeach
            </tbody>
        </table>
    </div>
@endsection
```

2. **Tampilan Create (form untuk membuat post baru)**: Buat file `resources/views/posts/create.blade.php`:

```
blade

@extends('layouts.app')

@section('content')
    <div class="container">
        <h1>Create Post</h1>
        <form method="POST" action="{{ route('posts.store') }}">
            @csrf
            <div class="mb-3">
                <label for="title" class="form-label">Title</label>
```

```
                <input type="text" class="form-control" name="title"
required>
            </div>
            <div class="mb-3">
                <label for="content" class="form-label">Content</label>
                <textarea class="form-control" name="content"
required></textarea>
            </div>
            <button type="submit" class="btn btn-
primary">Submit</button>
        </form>
    </div>
@endsection
```

3. **Tampilan Edit (form untuk mengedit post)**: Buat file `resources/views/posts/edit.blade.php`:

```
blade

@extends('layouts.app')

@section('content')
    <div class="container">
        <h1>Edit Post</h1>
        <form method="POST" action="{{ route('posts.update', $post->id)
}}">
            @csrf
            @method('PUT')
            <div class="mb-3">
                <label for="title" class="form-label">Title</label>
                <input type="text" class="form-control" name="title"
value="{{ $post->title }}" required>
            </div>
            <div class="mb-3">
                <label for="content" class="form-label">Content</label>
                <textarea class="form-control" name="content"
required>{{ $post->content }}</textarea>
            </div>
            <button type="submit" class="btn btn-
primary">Update</button>
        </form>
    </div>
@endsection
```

## Menjalankan Aplikasi

Setelah semua langkah di atas selesai, jalankan aplikasi Laravel Anda menggunakan perintah:

```
bash

php artisan serve
```

Akses aplikasi di browser melalui `http://localhost:8000/posts`. Anda sekarang dapat membuat, melihat, mengedit, dan menghapus post.

## Kesimpulan

Dalam modul ini, Anda telah belajar cara membangun aplikasi CRUD dasar menggunakan Laravel. Anda telah melakukan langkah-langkah untuk membuat tabel database, model Eloquent, controller, rute, dan tampilan. Laravel menyediakan struktur yang jelas dan alat yang efisien untuk mengelola data dalam aplikasi web. Dengan pemahaman dasar ini, Anda dapat mulai mengembangkan fitur lebih lanjut dan menyesuaikan aplikasi sesuai dengan kebutuhan Anda.