PENGENALAN SERVICE REPOSITORY

Bagus Mantonafi

Apa Itu Design Pattern Pada Pemrograman?

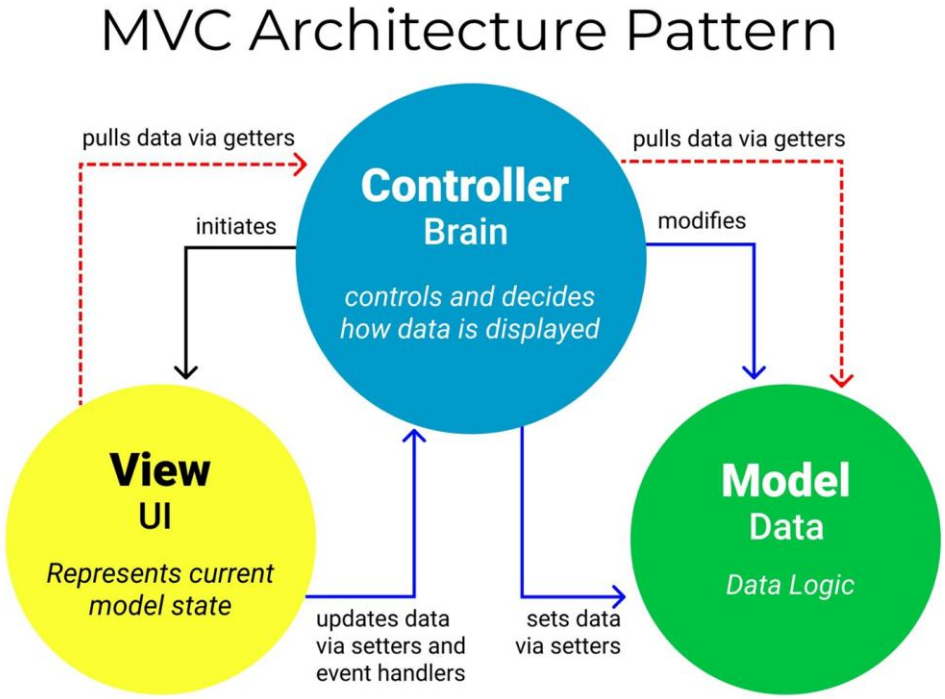
Design pattern adalah solusi umum untuk mengatasi permasalahan-permasalahan yang biasa terjadi ketika membuat Software.

Penggunaan design pattern dapat di Implementasikan ke bahasa pemrograman apa saja.

Manfaat Design Pattern?

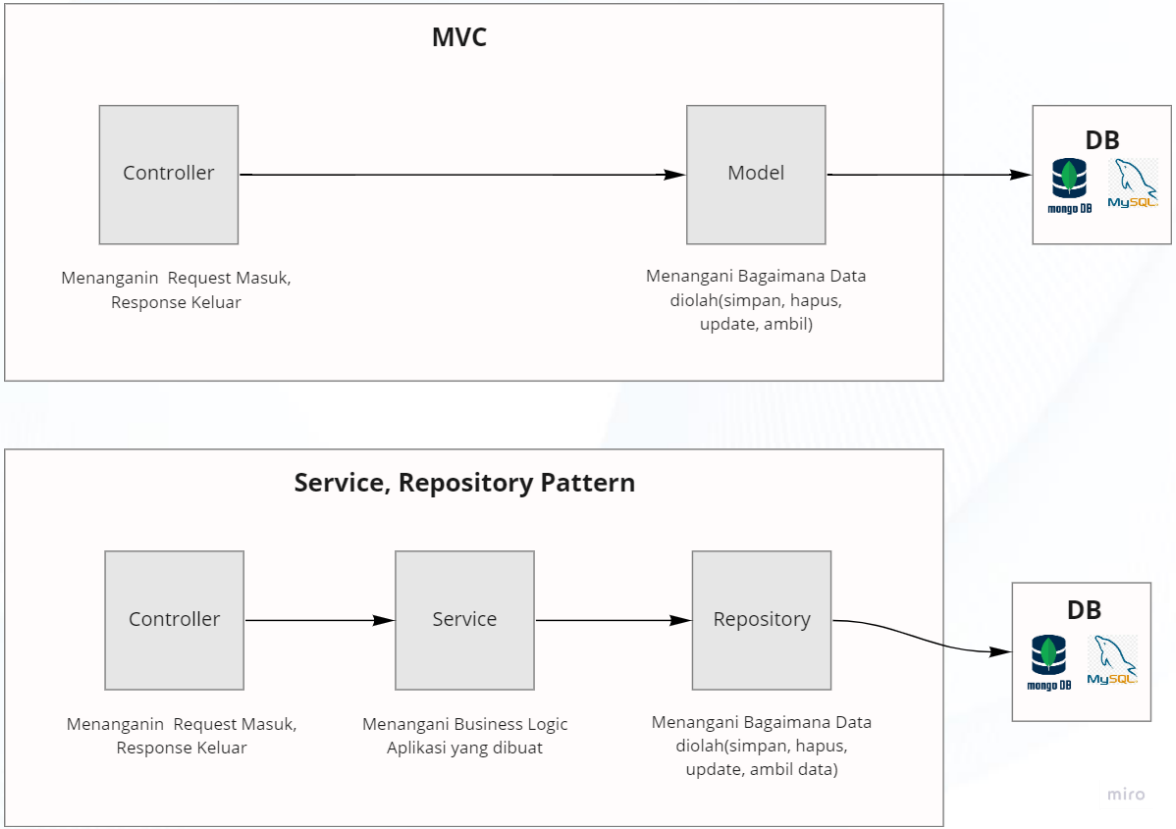
- Design Software lebih baik karena rapi, terstruktur
- Kode mudah dimengerti dan dirawat
- komunikasi antar team pengembang menjadi efisien

Design Pattern Pada Laravel (MVC)



Sumber :
freecodecamp.com

Laravel MVC & Service Repository Pattern



Service Layer / Facade Pattern?

- Service Layer adalah design pattern yang memisahkan business logic yang dikelompokan di class yang sama. Tujuannya adalah adalah agar logika yang telah dibuat dapat dipergunakan kembali tanpa harus menulis ulang.
- Service Layer Pattern sesuai dengan Prinsip DRY(Don’t Repeat Yourself)
  DRY sesuai dengan prinsip S.O.L.I.D yang pertama yakni Single Responsibility Principle (Class hanya memiliki satu tugas)

Service Layer / Facade Pattern

Business Logic pada Service adalah aturan bisnis di dunia nyata yang diimplementasikan kedalam program.

Contoh :

- Operasi Perhitungan seperti total transaksi, diskon, cashback, dll
- Akses Pengguna terhadap Data

Dan masih banyak contohnya

```php
class UserService {
    public function isUserValidate($user) {
        // Misal ada logika tambahan disini
    }

    public function handleLogin($email, $pass) {
        $user = $this->getUser( email: $email, password: $pass);
        if($this->isUserValidate( user: $user)) {
            return true;
        } else {
            return false;
        }
    }

    public function getUser($email, $password) {
        $user = User::where([
            'email'=>$email,
            'password'=>$password
        ])->first();
        return $user;
    }
```

Service yang dibuat

Repository Pattern?

- “Menggabungkan segala hal yang berhubungan dengan manipulasi database di satu tempat”
- Memudahkan ketika ingin berganti database
- Memisahkan logika dengan akses ke database
- Memudahkan ketika melakukan mocking data ketika melakukan pengujian

```php
class UserRepository
{
    public function getUser($email, $password) {
        $user = User::where([
            'email'=>$email,
            'password'=>$password
        ])->first();
        return $user;
    }

    public function getAllUser() {}
    public function getUserById($id) {}
    public function getUserWhoLikes() {}
    public function saveUser($databaru) {}
    public function deleteUser($id) {}
    public function softDeleteUser($id) {}
}
```

Repository berisi fungsi ambil data user dari Model

Implementasi Service Repository Pattern Pada Laravel

Aplikasi TODO sederhana : Implementasi 1

Membuat Task Baru

TaskController.php (tanpa Service-Repository Pattern)

```php
public function createTask(Request $request)
{
    $request->validate([
        'title'=>'required|string|min:3',
        'description'=>'required|string'
    ]);

    $data = [
        'title'=>$request->post('title'),
        'description'=>$request->post('description')
    ];

    $dataSaved = [
        'title'=>$data['title'],
        'description'=>$data['description'],
        'assigned'=>null,
        'subtasks'=> [],
        'created_at'=>time()
    ];

    $id = $this->tasks->save($dataSaved);

    $task = $this->tasks->find(['_id'=>$id]);

    return response()->json($task);
}
```

Membuat Task Baru(Service-Repository)

TaskController.php

```php
public function createTask(Request $request)
{
    $request->validate([
        'title'=>'required|string|min:3',
        'description'=>'required|string'
    ]);

    $data = [
        'title'=>$request->post('title'),
        'description'=>$request->post('description')
    ];;

    $taskId = $this->taskService->addTask($data);
    $task = $this->taskService->getById($taskId);
    return response()->json($task);
}
```

TaskService.php

```php
/**
 * NOTE: menambahkan task
 */
public function addTask(array $data)
{
    $taskId = $this->taskRepository->create($data);
    return $taskId;
}
```

TaskRepository.php

```php
/**
 * Untuk membuat task
 */
public function create(array $data)
{
    $dataSaved = [
        'title'=>$data['title'],
        'description'=>$data['description'],
        'assigned'=>null,
        'subtasks'=> [],
        'created_at'=>time()
    ];

    $id = $this->tasks->save($dataSaved);
    return $id;
}
```

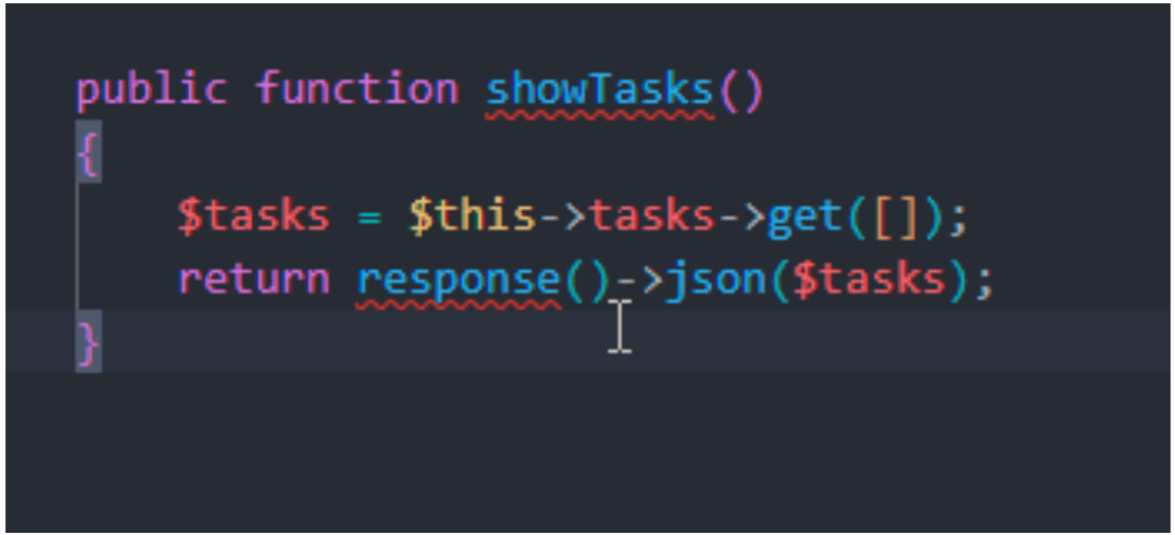
miro

Implementasi Service Repository Pattern Pada Laravel

Aplikasi TODO sederhana : Implementasi 2

Menampilkan Semua Tasks

TaskController.php (tanpa Service-Repository Pattern)



miro

Menampilkan semua task(Service Repository)

TaskController.php

```php
public function showTasks()
{
    $tasks = $this->taskService->getTasks();
    return response()->json($tasks);
}
```

TaskService.php

```php
/**
 * NOTE: untuk mengambil semua tasks di collection task
 */
public function getTasks()
{
    $tasks = $this->taskRepository->getAll();
    return $tasks;
}

/**
```

TaskRepository.php

```php
/**
 * Untuk mengambil semua tasks
 */
public function getAll()
{
    $tasks = $this->tasks->get([]);
    return $tasks;
}
```

miro

Terima Kasih

Referensi

https://www.freecodecamp.org/news/the-model-view-controller-pattern-mvc-archite cture-and-frameworks-explained/
https://www.youtube.com/watch?v=zD-bTMojnaM
https://www.youtube.com/watch?v=DKMT1HStrD0&list=PL-CtdCApEFH_yiziXrQeo gYOJzCmD8XLM&index=10
https://www.youtube.com/watch?v=1oTBT3HrE78&list=PL-CtdCApEFH_yiziXrQeo gYOJzCmD8XLM&index=9
https://www.anbidev.com/prinsip-solid/