No.: Date.:

Nama : Abira Husnia
NIM : 121140113
kelas : Pemweb RA

Design Pattern

Solusi umum untuk masalah yang biasa terjadi dalam desain perangkat lunak. Design pattern bukan termasuk metode yang iangsvng dapat diubah menjadi kode program melainkan sebuah pola dasar atau template dari cara untuk menyelesaikan masalah tersebut. Design pattern membantu mempercepat pengembangan perangkat lunak karena pola-pola yang dijelaskan merupakan paradigma-paradigma yang telah teruji kegunaannya.

Jenis-jenis Design Pattern

- Creational Pattern (Pola Pembuatan)
  Solusi untuk menyelesaikan masalah yang ditemui dalam pembuatan suatu instance / objek. ex: factory method, prototype, builder
- Structural Pattern (Pola Struktural)
  Solusi untuk menyelesaikan masalah yang ditemui dalam pengaturan komposisi class dan objek. ex: adapter, composite, decorator, bridge
- Behavioral pattern (Pola Perilaku)
  Solusi untuk menyelesaikan masalah yang ditemui dalam komunikasi antar objek-objek. ex: template method, mediator, state, visitor

keuntungan Menggunakan Design Pattern

- Aplikasi menjadi lebih mudah dipelihara, dikembangkan, teruji, fleksibel,
- Memberikan solusi yang sudah teruji untuk menyelesaikan masalah yang muncul
- Membuat komunikasi antara tim pengembang menjadi lebih episien

KIKY

No.: Date.:

Konsep MVC (Model-View-Controller) Design Pattern

↳ Sebuah pola desain arsitektur dalam pengembangan perangkat lunak yang memisahkan kode program menjadi 3 bagian utama, yaitu:
- Model, View, dan Controller. Konsep MVC banyak diterapkan dalam berbagai framework atau platform pengembangan perangkat lunak

komponen MVC:
- Model: bertanggung jawab untuk mengelola dan berinteraksi dengan data yang disimpan dalam database / sumber lainnya. Dapat melakukan operasi menambah data, mengubah data, menghapus data, mencari data, dan menampilkan data. Model tidak sama dengan database
- View: menampilkan informasi / tampilan antarmuka kepada pengguna. Dapat menampilkan data dari berbagai sumber, termasuk model, database, API
- Controller: menghubungkan / mengatur model dan view dalam setiap permintaan / aksi dari pengguna

Contoh Kode

- Model

```
class User                          3 properti : id, name, email
{                                   id → menyimpan ID pengguna
    protected $table = 'users';     name → menyimpan nama pengguna
                                    email → menyimpan alamat email pengguna

    protected $fillable = [
        'name'
        'email'
        'password'
        'created_at'
        'updated_at'
    ];
}
```

KIKY

No.: Date.:

-View

```
<? php

$users = User::all();   → mendapatkan data dari model

echo '<h1> Daftar User </h1>'; → menampilkan judul

foreach ($users as $user) {
    echo '<p> Nama: '. $user->name. '</p>';
    echo '<p> Email: '. $user->email. '</p>';
}

?>
```

} menampilkan daftar semua user

-Controller

```
<? php

class UserController   → mendefinisikan kelas UserController
{
    protected $UserModel;
    public function __construct()
    {
        $this->userModel = new UserModel();   → menyimpan objek dari kelas
    }                                            UserModel
    public function index()
    {
        $users = $this->userModel->getAllUsers();
    }

?>
```

KIKY