Nama : Gaberia Sinaga
NIM : 12114012S
Kelas : Pemweb (RA) Senin, 04/12/2023

Design pattern

Design pattern adalah suatu solusi umum untuk masalah desain perangkat lunak yang sering muncul. Mereka merupakan pedoman atau template yang dapat digunakan untuk memecahkan masalah desain dengan cara yang terorganisir dan efisien. Design pattern membantu para pengembang perangkat lunak untuk mengatasi kompleksitas sistem dengan menyediakan struktur yang teruji dan terbukti secara konseptual. Dengan menggunakan design pattern, pengembang dapat menghindari membuat desain yang bermasalah atau tidak efisien, serta mempromosikan prinsip-prinsip pengembangan perangkat lunak yang baik. Desain pattern dapat diterapkan dalam berbagai bahasa pemrograman dan lingkungan pengembangan, membuatnya menjadi alat yang sangat berharga dalam membangun sistem perangkat lunak yang handal dan mudah dipelihara. Ada 3 pola utama dalam design Pattern :

1. Creational patterns

Creational patterns dalam design pattern adalah kategori pola desain yang fokus pada proses pembuatan objek dan bagaimana menginstansiasi kelas atau objek-objek tersebut. Tujuan dari creational patterns adalah menyembunyikan kompleksitas pembuatan objek, mempromosikan fleksibilitas dalam instansiasi, dan memastikan bahwa sistem dapat beradaptasi dengan perubahan pada tahap penciptaan objek. Dengan menerapkan creational patterns, pengembang dapat meningkatkan fleksibilitas dan pemeliharaan kode, serta mengurangi ketergantungan langsung antara klien dan kelas konkret pada tahap penciptaan objek.

2. Behavioral patterns

Behavioral patterns dalam design pattern adalah kumpulan pola desain yang fokus pada cara objek-objek berinteraksi dan berkomunikasi satu sama lain. Pola-pola ini membantu mengatur alur kontrol dan tanggapan di antara objek-objek tersebut, sehingga sistem dapat berperilaku dengan cara yang terorganisir dan fleksibel.

3. Structural patterns

Structural patterns dalam design pattern adalah kategori pola desain yang berfokus pada cara komponen-komponen sistem dapat disusun dan dihubungkan untuk membentuk struktur yang lebih besar dan kompleks. Pola desain struktural membantu dalam mengorganisir kelas-kelas dan objek-objek agar dapat bekerja bersama secara efisien, serta memfasilitasi fleksibilitas dan perubahan dalam sistem. Beberapa contoh pola desain struktural meliputi Adapter Pattern, Bridge Pattern, dan Composite Pattern, yang masing-masing memberikan solusi untuk menangani masalah struktural dalam pengembangan perangkat lunak dengan memberikan cara yang lebih sistematis dan modular untuk menggabungkan komponen-komponen.

Senin, 04/12/2023

Konsep MVC (Model-View-Controller)
MVC adalah sebuah pola arsitektur perangkat lunak yang digunakan dalam pengembangan aplikasi untuk memisahkan komponen utama dari sistem agar dapat dielola secara terpisah. Konsep ini terdiri dari tiga komponen utama:

1. Model (Model)
Representasi data dan aturan bisnis dari aplikasi. Model bertanggung jawab untuk mengelola logika bisnis, melakukan manipulasi data, dan memberikan antarmuka untuk berkomunikasi dengan data tersebut. Model tidak memiliki pengetahuan tentang tampilan atau penggunaan tertentu yang dilakukan pada data.

2. View (Tampilan)
Bertanggung jawab untuk menampilkan data dari model kepada pengguna dan menerima input dari mereka. Tampilan mewakili antarmuka pengguna dan tidak memiliki pengetahuan tentang logika bisnis atau manipulasi data. Tampilan hanya menampilkan informasi kepada pengguna dan meneruskan input pengguna ke kontroler.

3. Controller (Kontroler)
Berfungsi sebagai perantara antara model dan tampilan. Kontroler menangani input pengguna, memproses logika bisnis yang sesuai, dan memperbarui model serta tampilan secara bersamaan. Kontroler memungkinkan pemisahan antara tampilan dan model sehingga perubahan dalam salah satu komponen tidak secara langsung memengaruhi yang lain.

Dengan memisahkan tugas-tugas ini, pola MVC memudahkan pengembangan aplikasi dengan memungkinkan perubahan dalam satu komponen tanpa harus memodifikasi yang lain. Ini juga memfasilitasi pengujian perangkat lunak karena masing-masing komponen dapat diuji secara terpisah. MVC digunakan secara luas dalam pengembangan perangkat lunak berbasis web dan desktop.

contoh code Model :

```
Class Admin
    def __init__(self, username, password, email):
        self.username = username
        self.password = password
        self.email = email

    def display_info(self):
        print(f"Admin Information: Username - {self.username}, Email - {self.email}")
```

Senin, 04/12/2023

```
  def change_password(self, new_password):
        self.password = new_password
        print("Password updated successfully.")

# Contoh penggunaan
admin1 = Admin("admin_user", "admin_pass123", "admin@example.com")
admin1.display_info()

admin1.change_password("new_password123")
admin1.display_info()
```

Penjelasan code :
Setiap objek Admin memiliki atribut seperti username, password, dan email, yang diinisialisasi melalui metode konstruktor __init__. Metode display_info digunakan untuk menampilkan informasi admin, termasuk nama pengguna (username) dan alamat email (email). Selain itu, terdapat metode change_password yang memungkinkan untuk mengganti password admin dengan nilai yang baru. Dalam contoh penggunaan, objek admin1 dibuat, informasinya ditampilkan, dan passwordnya diubah untuk menciptakan interaksi sederhana dengan model Admin.

Contoh code View :

```
<?php
class AdminView {
    public function displayAdminInfo($adminName, $adminRole) {
        echo "<h2>Admin Information</h2>";
        echo "<p><strong>Name:</strong> $adminName</p>";
        echo "<p><strong>Role:</strong> $adminRole</p>"; }
    public function displayWelcomeMessage() {
        echo "<p>Welcome, Admin!</p>";
    }
    public function displayUserList($userList) {
        echo "<h2>User List</h2>";
        echo "<ul>";
        foreach ($userList as $user) {
            echo "<li>$user</li>";
        }
        echo "</ul>"; }
?>
```

Senin, 04/12/2023

Penjelasan Kode :

Kode tersebut menunjukkan merupakan contoh sederhana dari sebuah kelas View untuk entitas "Admin" dalam pola arsitektur Model-View-Controller (MVC). Kelas AdminView memiliki beberapa metode untuk menampilkan informasi admin, pesan selamat datang, dan daftar pengguna. Metode displayAdminInfo menerima dua parameter, yaitu nama admin dan peran admin, dan menampilkan informasi tersebut dalam format HTML. Metode displayWelcomeMessage menampilkan pesan selamat datang kepada admin, sementara displayUserList menerima daftar pengguna sebagai parameter dan menampilkannya dalam bentuk daftar HTML.

Contoh kode Controller :

```
<?php
class AdminController {
    private $model;
    public function __construct($model) {
        $this->model = $model;
    }
    public function updateUserProfile($userId, $newData) {
        $user = $this->model->getUserById($userId);
        $updateduser = $this->updateUserData($user, $newData);
        $this->model->updateUser($userId, $updateduser);
        return "Profil pengguna berhasil diperbarui";
    }
    private function updateUserData($user, $newData) {
        $user['username'] = $newData['username'];
        $user['email'] = $newData['email'];
        return $user;
    }}
?>
```

Penjelasan Code :

Kelas `AdminController` dalam kode berperan sebagai kontroler. Memiliki tanggung jawab untuk mengelola tindakan administratif, khususnya memperbarui profil pengguna. Dalam konstruktor, objek model disertakan untuk berinteraksi dengan data pengguna. Metode `updateUserProfile` menerima ID pengguna dan data baru, kemudian menggunakan model untuk mengambil dan memperbarui data pengguna. Logika bisnis terkait pembaruan data terdapat dalam metode pribadi `updateUserData`. Keseluruhan struktur mencerminkan pemisahan peran dalam pola MVC, di mana kontroler mengelola logika aplikasi, berinteraksi dengan model, dan menyediakan respons untuk tampilan.