

PHP – MVC

1. Apa itu MVC?

MVC (Model-View-Controller) adalah sebuah pola arsitektur perangkat lunak yang digunakan untuk memisahkan logika aplikasi menjadi tiga bagian utama, yaitu Model, View, dan Controller. Ini membantu dalam pengembangan aplikasi yang lebih terstruktur, modular, dan mudah dikelola, terutama dalam pengembangan aplikasi berbasis web.

Berikut adalah penjelasan singkat tentang ketiga komponen dalam konsep MVC:

a. Model:

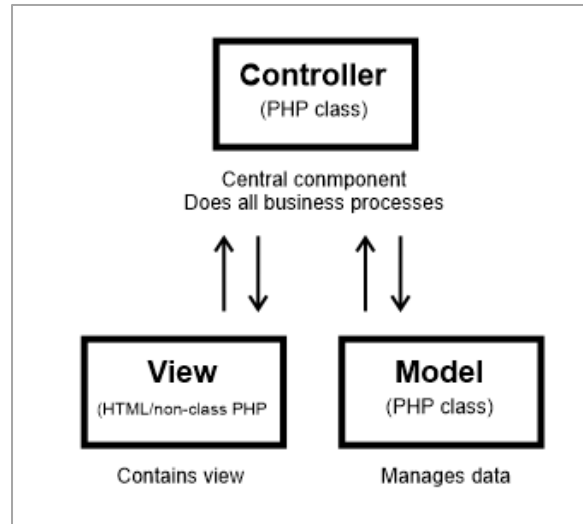
- Model berhubungan dengan semua data dan logika bisnis dari aplikasi.
- Model bertanggung jawab untuk mengelola data, termasuk mengakses basis data, memproses, dan memvalidasi data, serta memastikan integritas data.
- Model juga menyediakan data untuk View atau Controller sesuai dengan permintaan.
- Dalam konteks web, Model biasanya berisi logika untuk interaksi dengan database, seperti CRUD (Create, Read, Update, Delete) operasi.

b. View:

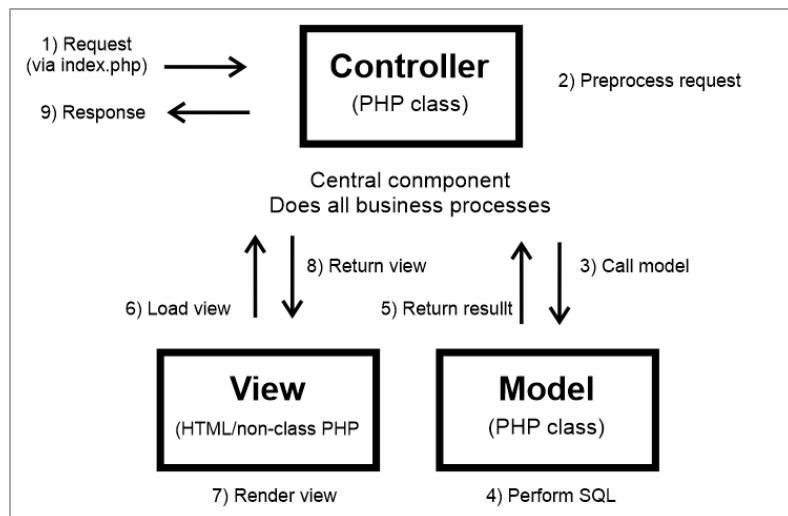
- View adalah bagian yang bertanggung jawab untuk menampilkan antarmuka pengguna (UI).
- View menampilkan data yang disediakan oleh Model kepada pengguna.
- View tidak memiliki logika bisnis. Perannya terbatas pada penyajian data, misalnya dalam bentuk HTML, JSON, XML, atau format lainnya.
- Setiap perubahan dalam View biasanya dipicu oleh Controller, bukan oleh Model secara langsung.

c. Controller:

- Controller bertindak sebagai penghubung antara Model dan View.
- Controller menerima input dari pengguna (misalnya melalui form atau permintaan HTTP), memproses input tersebut dengan berinteraksi dengan Model, dan kemudian memperbarui View.
- Controller juga menentukan bagaimana data dari Model ditampilkan di View dan bagaimana aplikasi merespons tindakan pengguna.
- Cara Kerja MVC:



Gambar 1 Hubungan komponen pada MVC (Konsep MVC)

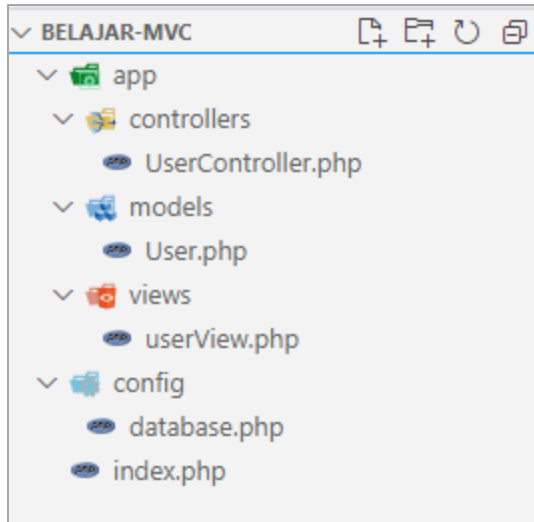


Gambar 2 Alur proses pada MVC

2. Membuat program dengan konsep MVC

a. Membuat struktur proyek dengan nama **belajar-mvc** lalu buat struktur sbb:

```
.
├── index.php          # Entry point aplikasi (Controller utama)
├── app
│   ├── controllers    # Folder berisi semua controller
│   │   └── UserController.php
│   ├── models         # Folder berisi semua model
│   │   └── User.php
│   └── views          # Folder berisi semua tampilan (view)
│       └── userView.php
└── config
    └── database.php    # Konfigurasi database
```



b. Membuat komponen mvc

1) Model (app/models/User.php)

Model bertanggung jawab untuk menangani data dan logika bisnis. Model akan berinteraksi dengan database.

```
app > models > User.php > ...
1  <?php
2  class User
3  {
4      private $db;
5
6      public function __construct($dbConnection)
7      {
8          $this->db = $dbConnection;
9      }
10
11     public function getUserById($id)
12     {
13         $stmt = $this->db->prepare("SELECT * FROM users WHERE id = :id");
14         $stmt->bindParam(':id', $id);
15         $stmt->execute();
16         return $stmt->fetch(PDO::FETCH_ASSOC);
17     }
18 }
```

Penjelasan:

- Kelas User bertanggung jawab untuk berinteraksi dengan database dan melakukan operasi yang berhubungan dengan data pengguna.
- Fungsi getUserById() mengambil data pengguna berdasarkan ID dari database.

2) View (app/views/userView.php)

View bertanggung jawab untuk menampilkan data kepada pengguna.

Pemrograman Berbasis Web 2

```
app > views > userView.php > html
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5  |   <title>User Information</title>
6  </head>
7
8  <body>
9  |   <h1>User Details</h1>
10 |   <p>ID: <?php echo $user['id']; ?></p>
11 |   <p>Name: <?php echo $user['name']; ?></p>
12 |   <p>Email: <?php echo $user['email']; ?></p>
13 </body>
14
15 </html>
```

Penjelasan:

- File ini adalah bagian View yang hanya bertugas menampilkan informasi data pengguna yang dikirim dari Controller.
- Data pengguna disajikan dalam bentuk HTML.

3) Controller (app/controllers/UserController.php)

Controller bertanggung jawab untuk menerima permintaan dari pengguna, memprosesnya, dan memutuskan apa yang harus ditampilkan.

```
app > controllers > UserController.php > ...
1  <?php
2  require_once 'app/models/User.php';
3
4  class UserController
5  {
6  |   private $userModel;
7
8  |   public function __construct($dbConnection)
9  |   {
10 |       $this->userModel = new User($dbConnection);
11 |   }
12
13 |   public function show($id)
14 |   {
15 |       // Mengambil data pengguna dari model
16 |       $user = $this->userModel->getUserById($id);
17
18 |       // Memuat view dan meneruskan data pengguna
19 |       require_once 'app/views/userView.php';
20 |   }
21 }
```

Penjelasan:

- UserController bertanggung jawab atas proses logika dari permintaan pengguna.
- Fungsi show(\$id) menerima parameter ID, mengambil data pengguna dari model User, dan kemudian meneruskan data tersebut ke view (userView.php).

4) Database Configuration (config/database.php)

Mengatur koneksi database menggunakan PDO (PHP Data Objects).

```
config > database.php > ...
1  <?php
2  function getDBConnection()
3  {
4      try {
5          $db = new PDO('mysql:host=localhost;dbname=dbmvc', 'root', '');
6          $db->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
7          return $db;
8      } catch (PDOException $e) {
9          echo 'Connection failed: ' . $e->getMessage();
10     }
11 }
```

Penjelasan:

- Fungsi `getDBConnection()` menyediakan koneksi ke database menggunakan PDO, yang digunakan oleh Model untuk mengakses database.

5) Main Entry Point (index.php)

File utama yang menangani routing dan memanggil Controller.

```
index.php > ...
1  <?php
2  require_once 'config/database.php';
3  require_once 'app/controllers/UserController.php';
4
5  // Koneksi ke database
6  $dbConnection = getDBConnection();
7
8  // Mendapatkan parameter dari URL (misalnya: index.php?id=1)
9  $id = isset($_GET['id']) ? intval($_GET['id']) : 1;
10
11 // Membuat instance UserController
12 $controller = new UserController($dbConnection);
13
14 // Menampilkan data pengguna berdasarkan ID
15 $controller->show($id);
16 ?>
```










Penjelasan:

- File `index.php` berfungsi sebagai entry point aplikasi.
- Ini menginisialisasi koneksi ke database, memuat Controller, dan memanggil metode `show()` pada `UserController` untuk menampilkan informasi pengguna berdasarkan ID yang diterima dari URL.

6) Membuat database

Buat database dengan nama **dbmvc**, lalu tambahkan sebuah tabel dengan nama **users** yang berisi terdiri dari tiga kolom (id, name dan email) kemudian tambahkan beberapa data ke dalam tabel tersebut

Pemrograman Berbasis Web 2

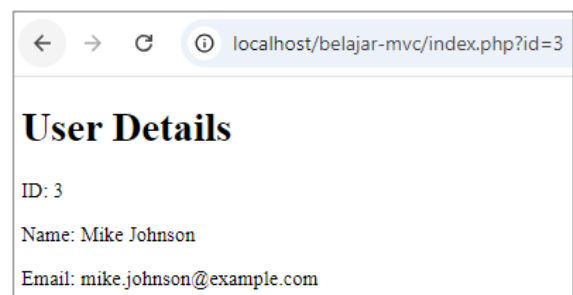
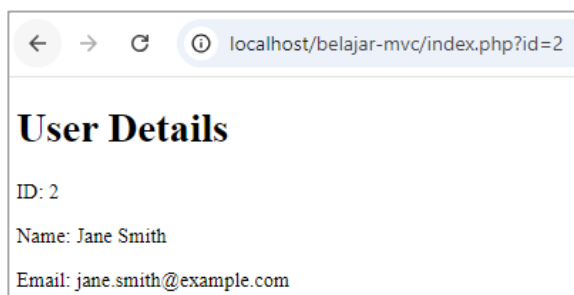
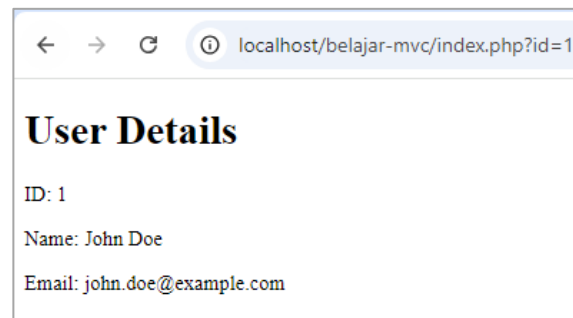
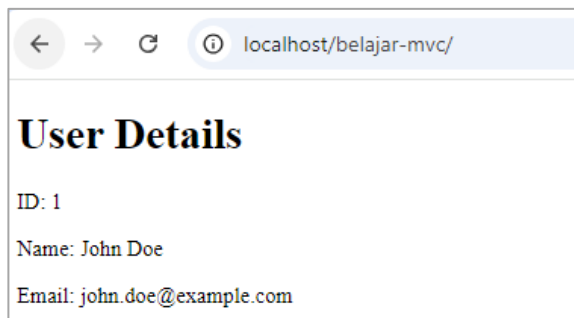
						id	name	email
<input type="checkbox"/>	 Ubah	 Salin	 Hapus	1	John Doe	john.doe@example.com		
<input type="checkbox"/>	 Ubah	 Salin	 Hapus	2	Jane Smith	jane.smith@example.com		
<input type="checkbox"/>	 Ubah	 Salin	 Hapus	3	Mike Johnson	mike.johnson@example.com		

7) Alur kerja program

- Pengguna mengakses URL seperti `http://localhost/belajar-mvc/index.php?id=1`. Parameter id dikirimkan melalui URL.
- File `index.php` menangkap parameter ID tersebut dan menginisialisasi Controller yang relevan (UserController).
- Controller (UserController) memanggil Model (User) untuk mengambil data dari database berdasarkan ID.
- Model melakukan query ke database dan mengembalikan data pengguna ke Controller.
- Controller kemudian memuat View (`userView.php`) dan meneruskan data pengguna ke View.
- View akan menampilkan data kepada pengguna dalam bentuk HTML.

Tampilan program

Ketika user mengakses url `localhost/belajar-mvc/` maka akan ditampilkan user dengan id 1, sama seperti ketika menambahkan `id=1`



TUGAS

Buat tampilan dengan menggunakan bootstrap. Dan jika pengguna mengakses `localhost/belajar-mvc/` (halaman index) maka akan menampilkan seluruh data users. Pada tampilan ini tambahkan kolom aksi untuk tombol detail, edit dan hapus. Dan jika pengguna mengeklik tombol detail maka akan tampil page informasi detail dari user.