

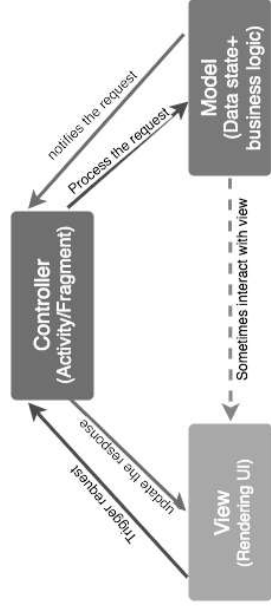


## Model-View-Controller

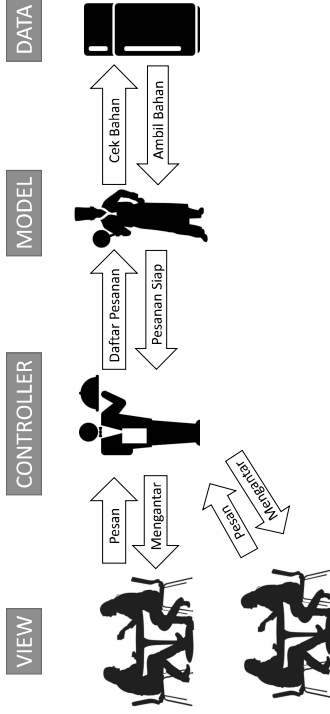
Guntur Maulana Zamroni  
Informatika  
Universitas Ahmad Dahlan

<https://tif.uad.ac.id/>

### Konsep MVC

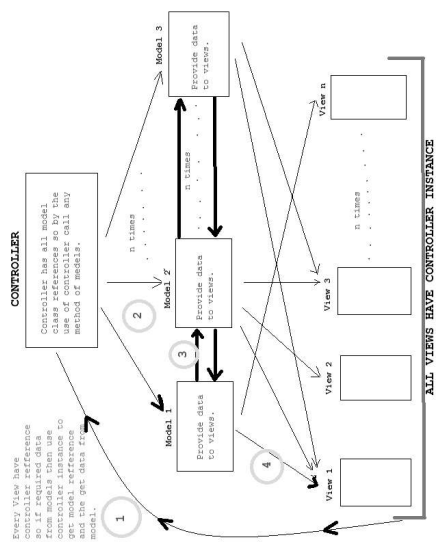


### Konsep MVC



### Pengertian

- Model-View-Controller atau MVC adalah sebuah metode untuk membuat sebuah aplikasi dengan memisahkan data (Model) dari tampilan (View) dan cara bagaimana memprosesnya (Controller).
- Konsep MVC diperkenalkan oleh Trygve Reenskaug pada tahun 1970an.
- Masing-masing layer (M-V-C) memiliki peran atau tugas ,dan hubungannya dengan layer lain.



### Bagian MVC

- **Model**  
Model mewakili struktur data. Biasanya model berisi fungsi-fungsi yang membantu seseorang dalam pengelolaan basis data seperti memasukkan data ke basis data, pembaruan data dan lain-lain.
- **View**  
View adalah bagian yang mengatur tampilan ke pengguna. Bisa di katakan berupa halaman aplikasi.
- **Controller**  
Controller merupakan bagian yang menjembatani model dan view. Controller berisi perintah-perintah yang berfungsi untuk memproses suatu data dan mengirimkannya ke halaman web.

- Secara sederhana:

Model → mengelola **data**, logic, dan rules pada sebuah aplikasi

Contoh: Content Provider, database, class yang menjalankan business logic

View → bagian **User Interface**

Contoh: XML Layout, resources, widget

Controller → **event handler**, menghubungkan View dan Model dengan merespon input user dan mengambil data sesuai permintaan user

Contoh: Activity, Fragment, Service

- Dengan menggunakan metode MVC maka aplikasi akan lebih mudah untuk dirawat dan dikembangkan.
- Untuk memahami metode pengembangan aplikasi menggunakan MVC diperlukan pengetahuan tentang pemrograman berorientasi objek (Object Oriented Programming).

## Contoh Skenario MVC – Aplikasi Kuis

- Pada sebuah aplikasi game quiz Android diketahui terdapat data sebagai berikut:

- Nama user
- Nilai sementara user
- List pertanyaan
- List jawaban
- List jawaban yang tepat

- Model kita gunakan untuk mengelola data yang berbeda-beda tersebut

- Pada skenario di atas, kita dapat membuat 2 class:

- 1 untuk mengelola **informasi user & nilai** → misal: class **User**
- 1 untuk mengelola **pertanyaan & jawaban** → misal: class **Question**

- Setelah Model selesai, lanjutkan dengan Controller. Semisal kita membuat class **GameActivity** yang bertugas untuk **menampilkan list pertanyaan** ke View (user).

- Pada skenario ini, tugas Controller adalah:

- Mengambil soal dari model dan 4 pilihan jawaban kemudian menampilkan ke user
- Mengambil pilihan jawaban user, verifikasi, dan menampilkan hasil apakah jawaban benar / salah
- Merekam nilai sementara user
- Menampilkan nilai akhir

- Lanjutkan dengan membuat View dan hubungkan dengan Controller

## Beberapa Konsep Pola Pengembangan Software

1. MVC → Model-View-Controller
2. MVP → Model-View-Presenter
3. MVVM → Model-View-View Model

