**2. Arsitektur Sistem**

**2.1 Pendekatan Arsitektur: Model-View-Controller (MVC)**

Sistem ini menggunakan arsitektur **Model-View-Controller (MVC)** untuk memisahkan tanggung jawab antara pengelolaan data, logika aplikasi, dan antarmuka pengguna. Pendekatan ini dipilih untuk mendukung modularitas, kemudahan pengembangan, serta perawatan sistem di masa depan.

**2.2 Komponen MVC**

**1. Model**

Model bertanggung jawab untuk menangani seluruh proses interaksi dengan basis data. Model akan merepresentasikan entitas dalam sistem seperti:

* Dosen (data profil, NIDN/NIP, keahlian)
* JadwalMengajar (data jadwal dosen)
* Evaluasi (data hasil evaluasi kinerja)

Model ini akan menangani validasi, relasi antar entitas, serta proses CRUD (Create, Read, Update, Delete).

**2. View**

View bertanggung jawab atas tampilan antarmuka pengguna (UI). Setiap halaman antarmuka dibangun secara dinamis berdasarkan data yang dikirimkan oleh Controller.  
Contoh tampilan:

* Halaman input data dosen
* Halaman daftar jadwal
* Formulir evaluasi dosen
* Grafik hasil evaluasi

**3. Controller**

Controller bertugas menjembatani View dan Model. Controller akan menerima permintaan (request) dari pengguna, mengolah logika bisnis, meminta data dari Model, dan mengirimkan respons ke View.  
Contoh controller:

* DosenController: menangani input, edit, hapus, dan tampilkan data dosen
* JadwalController: menangani jadwal mengajar
* EvaluasiController: mengelola input dan rekap evaluasi

**2.3 Ilustrasi Alur MVC**

Pengguna (Admin / Dosen / Mahasiswa)

↓

[ View ]

↓

[Controller]

↓

[ Model ]

↓

[Database]

Contoh alur:  
Mahasiswa → Mengisi formulir evaluasi → Controller menyimpan ke tabel evaluasi → View menampilkan notifikasi sukses.

**Diagram Arsitektur MVC (Visual)**

Berikut ini adalah diagram alur kerja sistem berdasarkan arsitektur MVC:

pgsql

CopyEdit

┌────────────────────┐

│ Pengguna │

│ (Admin/Dosen/Mhs) │

└────────┬───────────┘

↓

┌────────────┐

│ View │ ←─> HTML, CSS, JS, Template Engine

└────┬───────┘

↓

┌────────────┐

│ Controller │ ←─> Business Logic

└────┬───────┘

↓

┌────────────┐

│ Model │ ←─> Validasi, Query DB

└────┬───────┘

↓

┌────────────┐

│ Database │ ←─> MySQL / PostgreSQL

└────────────┘

**2.5 Contoh Integrasi MVC pada Fitur Sistem**

**1. Pengelolaan Data Dosen**

* **View**: Form tambah/edit dosen dan tabel data dosen.
* **Controller**: DosenController@store, DosenController@update.
* **Model**: Dosen.php (ORM), menyimpan data ke database.

**2. Penjadwalan Mengajar**

* **View**: Kalender/jadwal dosen per minggu.
* **Controller**: JadwalController@index, JadwalController@create.
* **Model**: JadwalMengajar.php, berelasi dengan Dosen.

**3. Evaluasi Kinerja Dosen**

* **View**: Formulir evaluasi untuk mahasiswa, grafik rekap evaluasi.
* **Controller**: EvaluasiController@store, EvaluasiController@rekap.
* **Model**: Evaluasi.php, menyimpan skor per indikator evaluasi.

**2.6 Keuntungan Arsitektur MVC dalam Sistem Ini**

* **Modularitas**: Setiap bagian sistem dapat dikembangkan terpisah.
* **Reusability**: Komponen seperti tampilan dan model bisa digunakan ulang.
* **Maintainability**: Lebih mudah dalam debugging dan scaling.
* **Team Collaboration**: Backend dan frontend bisa dikerjakan paralel.