**UML Data Mahasiswa**

**Use Case Diagram**

User

**Penjelasan :**

**Use Case** adalah sebuah kegiatan atau juga interaksi yang saling berkesinambungan antara aktor dan juga sistem. Atau dengan kata lain teknik secara umum digunakan, guna mengembangkan software / sistem informasi, guna memperoleh kebutuhan fungsional dari sistem yang ada. Komponen tersebut kemudian menjelaskan komunikasi antara actor,  dengan sistem yang ada.

* Aktor
* **User** dapat melakukan aktifitas ubah data, tambah data, hapus data, melihat data pada sistem.

**Actifity Diagram**

**1. Ubah Data 2. Hapus Data**

|  |  |
| --- | --- |
| **User** | **Sistem** |
| mulai  Mengubah data  Memilih menu ubah data | Menuju ke halaman ubah data      Update data yang sudah di ubah  Selesai |

|  |  |
| --- | --- |
| **User** | **Sistem** |
| mulai  Memilih data yang ingin di hapus  Memilih menu hapus data | Menuju ke halaman hapus data      Update data  Selesai |

|  |  |
| --- | --- |
| **User** | **Sistem** |
| mulai  Menambahkan data  Memilih menu tambah data | Menuju ke halaman tambah data      Update data  Selesai |

|  |  |
| --- | --- |
| **User** | **Sistem** |
| mulai  Memilih menu Lihat data | Menuju ke halaman Lihat data    Selesai |

**3. Tambah Data 4. Lihat Data**

|  |  |
| --- | --- |
| **User** | **Sistem** |
| mulai  Memilih menu keluar | Keluar    Selesai |

**5. Keluar**

**Penjelasan :**

**Activity Diagram** atau **Diagram aktivitas** adalah bentuk visual dari alur kerja yang berisi aktivitas dan tindakan, yang juga dapat berisi pilihan, atau pengulangan. Dalam Unified Modeling Language(UML), diagram aktivitas dibuat untuk menjelaskan aktivitas komputer maupun alur aktivitas dalam organisasi. Selain itu diagram aktivitas juga menggambarkan alur kontrol secara garis besar.

Kita ambil contoh **Actifity** ubah data.

Si aktor (user) melakukan pemilihan menu, ketika si user memilih ubah data maka sistem akan meresponnya kemudian sistem akan menampilkan atau mengarahkan ke halaman ubah data, kemudian user mulai mengubah data. Setelah user selesai mengubah data maka sistem akan secara otomatis menyimpan data tersebut kemudian kembali ke menu utama (selesai)

|  |
| --- |
| Operasi |
| - listData: List<data> |
| +addData(data: data) : void  +editData(data: data) : void  +removeData(data: data) : void  +getListData(): List<data> |

**Class Diagram**

|  |
| --- |
| Data |
| + NamaMahasiswa : String  + Alamat : String  + TempatTanggalLahir : String  + ProgramStudi : String  + nim : String  +no\_hp : String |
| +data()  +data(NamaMahasiswa, nim, Alamat, TempatTanggalLahir, ProgramStudi, no\_hp)  +setNamaMahasiswa(NamaMahasiswa): void  +getNamaMahasiswa(): String  +setnim(nim): void  +getnim(): String  +setAlamat(Alamat): void  +getAlamat(): String +setTempatTanggalLahir(TempatTanggalLahir):void  +getTempatTanggalLahir(): String  +setProgramStudi(ProgramStudi): void  +getProgramStudi(): String  +setno\_hp(no\_hp): void  +getno\_hp(): String |

use

1

**Penjelasan :**

**Class Diagram menggambarkan serta deskripsi atau penggambaran dari suatu class, atribut dan objek, disamping itu juga memiliki hubungan satu sama lain seperti pewarisan, asosiasi dan lainnya**

use

|  |
| --- |
| Aplikasi |
| - operasi: Operasi |
| + aplikasi()  +main(args:String[] ):void  +hapusData():void  +ubahData(): void  +listData(): void  +addData():void  +printMenu(): void |

**Sequence diagram**

**Sequence Diagram** adalah suatu diagram yang **menggambarkan interaksi objek** dan mengindikasikan (memberi petunjuk atau tanda) komunikasi diantara objek-objek tersebut . Sequence diagram digunakan untuk menggambarkan perilaku pada sebuah skenario dan mendeskripsikan bagaimana entitas dam sistem berinteraksi , termasuk pesan yang digunakan saat interaksi . Semua pesan dideskripsikan dalam urutan pada eksekusi . Sequence diagram berhubungan erat dengan Use Case Diagram , dimana **1 Use Case akan menjadi 1 Sequence Diagram**.

**Tambah data**

user

Sistem

1. Menambahkan data (User)

2. Menampilkan data

**Melihat Data**

User

Sistem

Tampilan Menu

2. Memilih menu tampilkanData()

3. menampilkan Data Mahasiswa()

1. Masuk ke sistem()