

Modul Praktikum

Analisis dan Desain Sistem Berorientasi Objek

Ayu Ratna Juwita, M.Kom

Cici Emilia Sukmawati, M.kom

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh.

Segala puji syukur kita panjatkan kepada Tuhan Yang Maha Esa yang telah memberikan berkah sehingga penerbitan Modul Praktikum Analisis dan Desain Sistem Berorientasi Objek Semester Ganjil 2023/2024 dapat terlaksana dengan baik. Penerbitan modul ini bertujuan untuk memberikan panduan dan informasi kepada mahasiswa/i Jurusan Teknik Informatika, Univeristas Buana Perjuangan karawang dalam kegiatan praktikum.

Saya sangat berharap modul ini dapat membantu dan memandu mahsiswa/i dalam beraktivitas di Laboratorium Fakultas Ilmu Komputer. Terimakasih kepada pihak, yang berperan dalam penerbitan modul ini.

wassalamu'alaikum waahmahtullahi wabarakatuh.

Karawang, Agustus 2023

Penyusun

Ayu Ratna Juwita, M.Kom

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI.....	ii
MODUL 1 ANALISIS SISTEM	1
1.1. Analisis Sistem.....	1
1.2. Analisis dan Desain Berorientasi Objek.....	1
1.3. Analisis Kebutuhan Fungsional (<i>Functional Requirement</i>).....	3
1.4. Analisis Kebutuhan Non Fungsional (<i>Non Functional Requirement</i>)	3
1.5. Metodologi SDLC (<i>System Development Life Cycle</i>).....	4
1.6. Contoh Latihan.....	5
1.7. Tugas Praktikum 1	6
MODUL 2 OVERALL DESCRIPTION ANALYSIS	7
2.1. Introduction	7
2.2. Purpose	7
2.3. Scope	7
2.4. Definition, Acronyms, and Abbreviation	7
2.5. Product Perspective	8
2.6. Product Function	8
2.7. User Characteristic	8
2.8. Constraints.....	8
2.9. Assumptionts and Depedencies.....	9
2.10. Specific Requirements	9
2.11. Functionality	9
2.12. Hardware Interfaces.....	10
2.13. Performance Requirements.....	10
2.14. Software System Quality Attributes	10
2.15. Tugas Praktikum 2.....	10
MODUL 3 UNIFIED MODELLING LANGUAGE (UML).....	11
3.1. Pengenalan <i>Unified Modeling Language</i> (UML).....	11
3.2. Tipe Diagram UML.....	12
3.3. Simbol Diagram <i>Unified Modeling Language</i> (UML)	13
3.3.1. Usecase Diagram	13
3.3.2. <i>Class</i> Diagram.....	15

3.3.3. Activity Diagram	15
3.3.4. Sequence Diagram	16
3.4. Astah.....	17
3.5. Praktikum	18
3.6. Tugas Praktikum 3	19
MODUL 4 USECASE DIAGRAM	20
4.1. Dasar Teori.....	20
4.2. Contoh Latihan.....	22
4.3. Praktikum	28
4.4. Tugas Praktikum 4	29
4.5. Tugas Praktikum 5	29
MODUL 5 CLASS DIAGRAM	30
5.1. Class Diagram	30
5.2. Mendefinisikan Class Diagram	30
5.3. Cara mengidentifikasikan kelas	31
5.4. Nilai Kardinalitas	32
5.5. Contoh Class Diagram	32
5.6. Latihan Praktikum	33
5.7. Tugas Praktikum 6	33
5.8. Tugas Praktikum 7	33
5.9. Uji Pemahaman Materi.....	33
MODUL 6 ACTIVITY DIAGRAM	34
6.1. Pengertian Activity Diagram.....	34
6.2. Elemen-Element dari Actyvity Diagram.....	34
6.3. Contoh Aktivty Diagram.....	35
6.4. Latihan Praktikum	37
6.5. Tugas Praktikum 8	37
6.6. Tugas Praktikum 9	37
MODUL 7 SEQUENCE DIAGRAM.....	38
7.1. Sequence Diagram.....	38
7.2. Obyek/ Participant.....	38
7.3. Message.....	39
7.4. Time.....	39
7.5. Tujuan Sequence Diagam.....	39
7.6. Contoh Sequence Diagram.....	40

7.7. Latihan Praktikum	41
7.8. Tugas Praktikum 8	41
7.9. Tugas Praktikum 9	41
MODUL 8 USER INTERFACE/MOCKUP SYSTEM	42
8.1. User Experience	42
8.1.1. UX User	42
8.1.2. UX Admin.....	43
8.2. User Interface	44
8.2.1. UI User.....	44
8.2.2. UI Admin	45
8.3. Performance Aplikasi.....	46
MODUL 9 UJI PEMAHAMAN PRAKTIKUM.....	47
DAFTAR PUSTAKA.....	50

MODUL 1

ANALISIS SISTEM

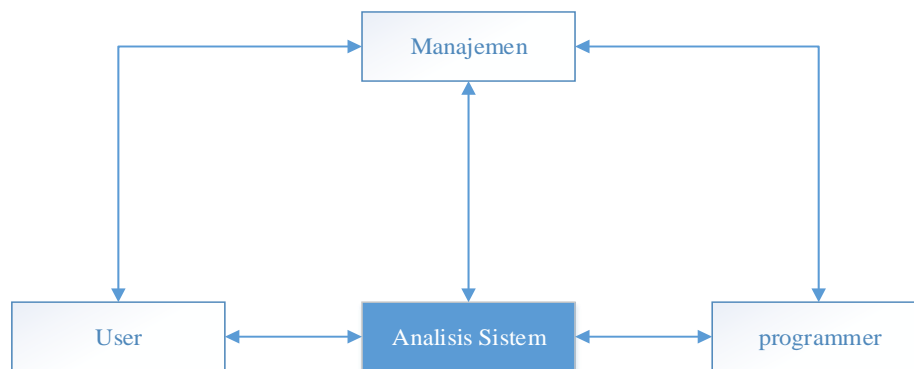
1.1. Analisis Sistem

Suatu analisis yang akan dirancang oleh suatu orang atau tim. Orang yang merancang system ini disebut dengan Analisis Sistem (System Analyst). Terdapat beberapa definisi yang mengidentifikasi Sistem Analisis:

1. Seorang yang menggunakan pengetahuan aplikasi komputer yang dimilikinya untuk memecahkan masalah-masalah bisnis, dibawah petunjuk Manajer Sistem.
2. Seorang yang bertanggung jawab menterjemahkan kebutuhan-kebutuhan yang menggunakan system (*user*) kedalam spesifikasi teknik yang diperlukan oleh programmer dan diawasi oleh Manajemen.

Adapun fungsi dari analisis system antara lain :

1. Mengidentifikasi masalah-masalah dari user
2. Menyatakan secara spesifik sasaran yang harus dicapai untuk memenuhi kebutuhan *user*
3. Memilih alternatif-alternatif metode pemecahan masalah
4. Merencanakan dan menerapkan rancangan sistemnya sesuai dengan permintaan *user*



Gambar 1. 1 *Ilustrasi Keberadaan Analisis Sistem*

1.2. Analisis dan Desain Berorientasi Objek

Analisis dan desain berorientasi objek adalah cara baru dalam memikirkan sesuatu masalah dengan menggunakan model yang dibuat menurut konsep sekitar dunia nyata. Dasar pembuatan adalah objek, yang merupakan kombinasi antara struktur data dan perilaku dalam suatu entitas (Aries Hadi Sutopo 2002). Berorientasi objek merupakan paradigma baru dalam

rekayasa perangkat lunak yang memandang system sebagai kumpulan objek-objek distrit yang salaing berinteraksi satu sama lain (Sholiq, Prof. Dr. Ir. Imam Robandi 2010).

Tahap analisis dilakukan setelah tahap perencanaan sistem dan sebelum perancangan sistem. Tahap ini merupakan tahap yang kritis dan sangat penting, karena kesalahan dalam tahapan ini menyebabkan kesalahan pada tahap selanjutnya. Terdapat empat langkah dasar untuk melakukan pembangunan dan pengembangan sistem diantara nya :

1. **Identify** (Indentifikasi masalah)
 - a. Mengidentifikasi penyebab masalah
 - b. Mengidentifikasi titik keputusan
 - c. Mengidentifikasi personil-personil kunci (bagan alur dokumen perusahaan serta deskripsi kerja (*job description*))
2. **Understand** (memahami kerja system yang ada)
 - a. Wawancara
 - b. Observasi
 - c. Daftar pertanyaan
 - d. Pengambilan sample

Tugas yang perlu dilakukan pada langkah ini yaitu :

- a) Menentukan jenis penelitian
 - b) Merencanakan jadwal penelitian
 - c) Mengatur jadwal penelitian
 - d) Mengatur jadwal observasi
 - e) Mengatur jadwal pengambilan sample
 - f) Membuat penugasan penelitian
 - g) Membuat agenda wawancara
 - h) Mengumpulkan hasil penelitian
3. **Analyze** (menganalisis sistem)

Data ini dilakukan berdasarkan data yang telah diperoleh dai hasil penelitian yang telah dilakukan.
4. **Report** (membuat laporan hasil analisis)

Data *report* ini diperlukan ketika melakukan analisis system, hasil laporan analisis akan di buat menggunakan *software requirement Spesificationi* agar mempermudah dikumentasi analisis sistem.

Tujuan dari analisis sistem antara lain :

- a. Menjabarkan kebutuhan pemakai
- b. Meletakkan dasar-dasar untuk proses perancangan perangkat lunak
- c. Mendefinisikan semua kebutuhan pemakai sesuai dengan lingkup kedua belah pihak.

1.3. Analisis Kebutuhan Fungsional (*Functional Requirement*)

Merupakan kegiatan dalam mendeskripsikan fungsionalitas atau layanan yang diharapkan akan diberikan oleh sistem. Persyaratan fungsional untuk sistem perangkat lunak bisa dinyatakan dalam sejumlah cara. Persyaratan tersebut bergantung pada jenis perangkat lunak, user yang menggunakan, dan jenis sistem yang akan dikembangkan.

Contoh :

Terdapat sejumlah persyaratan fungsional untuk sistem perpustakaan, bagi mahasiswa dan dosen untuk memesan buku dan dokumen dari perpustakaan lain:

1. User dapat mencari semua atau satu set awal database atau memilih subset darinya
2. Sistem akan menyediakan viewer yang sesuai bagi user untuk membaca dokumen pada penyimpanan (store) dokumen.

Pada prinsipnya spesifikasi persyaratan fungsional untuk sebuah sistem harus lengkap dan konsisten.

1.4. Analisis Kebutuhan Non Fungsional (*Non Functional Requirement*)

Merupakan persyaratan yang tidak langsung berhubungan dengan fungsi spesifik yang disediakan oleh sistem. Persyaratan ini mungkin berhubungan dengan properti sistem yang muncul belakangan seperti kehandalan, waktu tanggap, dan penempatan pada media penyimpanan. Alternatifnya, persyaratan ini dapat mendefinisikan batasan pada sistem seperti kemampuan piranti I/O dan representasi data yang dipakai pada *interface* sistem. Kebutuhan non fungsional ini meliputi kebutuhan :

a. *Development Requirement*

Tools yang digunakan (hardware dan software) untuk pengembangan sistem.

Contoh : eclipse, netbeans, starUML dsb.

b. *Deployment Requirement*

Terkait dengan lingkungan dimana sistem akan digunakan.

Contoh: sistem harus mampu berjalan dengan spesifikasi RAM 4GB, OS Ubuntu, dsb.

c. *Performance Requirement*

Terkait dengan ukuran kualitas maupun kuantitas khususnya terkait dengan kecepatan, skalabilitas, dan kapasitas.

Contoh: sistem harus mampu diakses oleh 100 orang dalam waktu bersamaan.

d. *Dokumentation Requirement*

Terkait dengan dokumen apa saja yang akan disertakan pada produk akhir.

Contoh : dokumen teknis (dokumen perencanaan proyek, analisis , desain, pengujian), user manual,dan dokumen pelatihan.

e. *Support Requirement*

Kebutuhan yang terkait dengan dukungan yang diberikan setelah sistem informasi digunakan.

Contoh: perlu adanya pelatihan bagi calon pengguna.

1.5. Metodologi SDLC (*System Development Life Cycle*)

Fokus utama dalam metodologi adalah objek. Suatu sistem dapat dilihat dari objek yang saling berhubungan. Objek dapat digambarkan sebagai benda, orang, tempat dan sebagainya yang mempunyai atribut dan metode. Metodologi terdiri dari pembuatan model dari domain aplikasi, kemudian menambahkan detail implementasi pada saat perancangan dari suatu system. Berikut tahapan dari metodologi *System Development Life Cycle* (SDLC).



Gambar 1. 2 Tahapan *Software Development Life Cycle* (SDLC)

Tahap-tahap metodologi berdasarkan *System Development Life Cycle* (SDLC) digunakan dengan memperhatikan karakteristik berorientasi objek yaitu : analisis, perancangan dan implementasi.

1.6. Contoh Latihan

SKENARIO :

Sebuah minimarket menjual berbagai macam barang dari mulai perelengkapan peralatan rumah tangga, alat tulis, dan bahan sembako yang dibutuhkan sehari-hari. Pemilik minimarket ingin membuat sistem yang bisa digunakan untuk transaksi jual beli. Sistem tersebut akan digunakan oleh pemilik minimarket sebagai direktur dan karyawannya sebagai kasir. Jika anda diminta untuk membangun sistem tersebut. Analisislah kebutuhan sistem terlebih dahulu.

Step 1 :

Identifikasi Objek (entitas) lengkap dengan atribut yang terlibat di dalam sistem tersebut.

Objek = direktur, barang, kasir

Step 2 :

Tuliskan jawaban anda kedalam model kamus data.

Direktur = {nama, alamat, no_telp}

Barang = {kode_barang, nama_barang, jenis, stok}

Kasir = {kode_kasir, nama, alamat, no_telp, jk}

Step 3 :

Tuliskan kebutuhan fungsional sistem

1. **Kasir dapat menggunakan sistem untuk transaksi penjualan**
2. **Sistem dapat digunakan untuk menyimpan data barang baru dan ketersediaan stok**
3. **Direktur dapat melihat laporan data barang dan data penjualan**

Step 4 :

Tuliskan kebutuhan non fungsional sistem

1. **Sistem ini berbasis desktop**
2. **Sistem di bangun dengan menggunakan bahasa pemrograman java**
3. **Sistem dapat berjalan pada komputer dengan spesifikasi minimal RAM 2GB**

1.7. Tugas Praktikum 1

1. Diketahui sebuah Apotek telah beroperasi selama 10 tahun. Selama ini apotek tersebut sebelumnya sudah menggunakan sistem informasi persediaan barang obat-obatan dan perlengkapan lainnya, Namun sistem ini pertama kali di implementasikan pada 6 tahun yang lalu, dan belum pernah dikembangkan kembali, pemilik apotek ingin mengupdate sistem informasi tersebut, dari segi sistem *front end* dan *Back end*, agar pembeli bisa melihat status ketersediaan barang yang dibutuhkan tanpa harus mengunjungi apotek terlebih dahulu, dan juga bisa memesan obat secara online. Dan dikarenakan teknologi pada saat ini sangatlah berperan dan berkembang sangat pesat pemilik apotek ingin mengupdate sistem tersebut.

Seorang Analis sistem hendak menganalisis sistem apotek tersebut, ia akan mengumpulkan informasi berkaitan dengan penggunaan dan per-*from* sistem informasi persediaan barang kepada para pengguna nya, yaitu pemilik apotek, karyawan dan pembeli.

- a. Tentukan objek data dan atribut pada yang bisa ditemukan pada kasus diatas, dan tuliskan hasil analisis kebutuhan fungsional dan nonfungsional untuk sistem informasi tersebut.
- b. Buatlah minimal 3 pertanyaan isi (inti) seputar sistem yang dimaksud menggunakan teknik wawancara. Responden Pemilik Apotek, karyawan dan Pembeli.
- c. Buatlah minimal 3 pertanyaan isi (inti) untuk teknik kuesioner. Responden adalah karyawan bagian Gudang dan kasir, bagian pembelian dan bagian keuangan.

MODUL 2

OVERALL DESCRIPTION ANALYSIS

2.1. Introduction

Dokumen *Software Requirements Specification (SRS)* ini memberikan gambaran tentang tujuan, ruang lingkup, definisi, singkatan-singkatan, referensi dan gambaran secara keseluruhan dari perangkat lunak.

2.2. Purpose

Tujuan dari dokumen ini adalah untuk mengumpulkan, menganalisis dan memberikan wawasan mendalam dari sistem pencarian rute terpendek menuju rumah sakit, lengkap dengan mendefinisikan pernyataan masalah secara rinci. Secara garis besar, dokumen ini menyajikan:

- a. Deskripsi tentang lingkungan produk yang akan digunakan.
- b. Deskripsi tentang kemampuan sistem.
- c. Persyaratan spesifikasi sistem yang digunakan untuk mengoperasikan produk sistem.

SRS ini memungkinkan dalam pemahaman tentang apa yang diharapkan dari sistem yang baru dan yang akan dibangun. Pemahaman yang jelas mengenai sistem dan fungsionalitas akan memungkinkan dikembangkannya produk yang tepat bagi pengguna. SRS ini dapat digunakan sebagai dasar dalam pengembangan proyek, dimana sistem pencarian rute terpendek menuju rumah sakit dapat dirancang, dibangun dan diuji.

Dokumen ini ditujukan pada pengguna dan pihak pengembang. Pembaca diasumsikan memiliki pengetahuan tentang sistem pencarian rute terpendek menuju rumah sakit serta pengetahuan dan pemahaman tentang *Unified Modeling Language (UML)* diagram.

2.3. Scope

Produk yang akan dibuat dalam dokumen ini merupakan bagian dari ruang lingkup kebutuhan pembangunan perangkat lunak yang berupa aplikasi yang digunakan untuk pengelolaan Sistem pencarian rute terpendek menuju rumah sakit yang dipilih oleh *user* pengguna.

2.4. Definition, Acronyms, and Abbreviation

SRS (Software Requirement Specification)	SRS adalah dokumen yang menjelaskan tentang kebutuhan fungsional maupun non-fungsional sistem perangkat lunak yang akan dikembangkan.
---	---

UML (Unified Modeling Language)	Bahasa spesifikasi standar untuk mendokumentasikan, menspesifikasikan, dan membangun sistem perangkat lunak.
DFD (Data Floew Diagram)	Data Flow Diagram atau DFD adalah langkah atau metode-metode yang digunakan untuk membuat perancangan sistem. Diagram ini dibuat dengan berorientasi pada alur data yang bergerak pada suatu sistem.
Shortest Path Greedy	Dalam sebuah Graph yang setiap edge yang memiliki weight (bobot), jarak terpendek (shortest path) antara 2 node dapat dicari dengan Metode Greedy

2.5. Product Perspective

Perangkat lunak sistem pencarian rute terpendek ini merupakan perangkat lunak yang digunakan untuk membantu masyarakat mempermudah pencarian rumah sakit dan rute terpendek menuju rumah sakit. Aplikasi ini berkaitan dengan entitas luar yaitu masyarakat umum. Sistem ini merupakan aplikasi informasi pencarian rute terpendek menggunakan algoritma *shortest path greedy*.

2.6. Product Function

Dengan adanya sistem pencarian rumah sakit dan rute terpendek menuju rumah sakit membantu *user* dengan mencari rute untuk menuju rumah sakit. Aplikasi ini juga mempunyai beberapa tampilan seperti adanya pilihan rumah sakit yang ingi dicari, informasi umum seputar rumah sakit.

2.7. User Characteristic

Dalam Rancang Bangun Sistem pencarian rute terpendek masing-masing *user* sebagai berikut :

1. admin

admin mempunyai hak untuk mengelola data rumah sakit.

2. user (pengguna sistem)

user menentukan titik awal atau lokasi *user* untuk melakukan pencarian rumah sakit dan mencari rute terpendek menuju rumah sakit.

2.8. Constraints

Batasan proyek sistem pencarian rute terpendek dalam dokumen SRS ini adalah sebagai berikut:

1. Sistem pencarian rute terpendek menuju rumah sakit yang dipilih.

2. Sistem pencarian rute terpendek ini akan diterapkan algoritma *shortest path greedy* untuk mendapatkan hasil rute terpendek

2.9. Assumptionts and Depedencies

Asumsi dan ketergantungan yang digunakan dalam proyek ini adalah:

1. Semua manajemen yang terkait dalam system pencarian rute terpendek ini hanya dapat dibuat dan dikelola oleh admin.
2. Tidak ada *trainning* program bagi *user* (*user* dianggap sudah mengerti dan dapat mengoperasikan program dengan baik).
3. Perangkat keras yang dibutuhkan untuk operasional program telah disediakan oleh pihak kampus.
4. Segala lisensi *software* ditanggung oleh pihak kampus.
5. Sistem Operasi yang digunakan minimal Windows 7.

2.10. Specific Requirements

Berikut adalah kebutuhan perangkat lunak untuk perancangan system dan petugas penguji dalam melakukan verifikasi Sehingga diperlukannya suatu pengolahan data-data yang diproses secara komputerisasi guna mendapatkan informasi-informasi yang berguna.

Bagian ini berisi semua persyaratan fungsional dan kualitas dari produk. Hal ini memberikan penjelasan secara rinci tentang sistem dan semua fitur-fiturnya.

2.11. Functionality

Tabel 2. 1 Fungsi sistem

No.	Fungsi	Deskripsi
1	Login	Proses login admin ke sistem pencarian rute rumah sakit
2	Input data rumah sakit	Proses untuk input data rumah sakit
3	Edit data rumah sakit	Proses untuk edit data rumah sakit
4	Hapus data rumah sakit	Proses untuk hapus data rumah sakit
5	Reset data rumah sakit	Proses untuk membersihkan form data rumah sakit
6	Bantuan	Fungsi untuk menampilkan bantuan sistem
7	Mengaktifkan layanan lokasi	Fungsi untuk mengaktifkan layanan lokasi
8	Tentukan posisi user	Fungsi untuk menentukan posisi user
9	tentang	Fungsi untuk menampilkan tentang informasi rumahsakit
10	Cari rumah sakit	Fungsi untuk pencarian rumah sakit
11	Pilih rumah sakit	Fungsi untuk memilih rumah sakit
12	Pilih rute	Fungsi untuk memilih rute terpendek menuju rumah sakit
13	View data Rumah Sakit	Fungsi untuk menampilkan list rumah sakit

2.12. Hardware Interfaces

Hardware Interface yang dibutuhkan untuk membantu kelengkapan dari pembangunan sistem yang sedang dirancang pada umumnya hanya berupa komputer.

Karena aplikasi ini tidak harus berjalan melalui internet, sehingga tidak perlu menggunakan internet. Dan koneksi dari komputer ke server database dikelola oleh sistem operasi yang mendukung phpmyadmin atau localhost dengan bantuan XAMPP.

2.13. Performance Requirements

Pada dasarnya kinerja dalam penggunaan produk akan tergantung pada komponen hardware dan koneksi *localhost* yang digunakan oleh *client/user*. Karena produk yang dihasilkan merupakan aplikasi desktop dan website. Produk ini memerlukan waktu pada saat memuat halaman awal tergantung pada kecepatan komputer pada saat dijalankan.

2.14. Software System Quality Attributes

Persyaratan di bagian ini menentukan reliability, availability, security dan maintainability yang diperlukan software system.

- *Usability* : Untuk mengakses sistem ini, user dapat menggunakan aplikasi dekstop sistem pencarian rute terpendek menuju rumah sakit. Sistem dapat di akses selama 7 X 24 jam, kecuali saat maintenance/ perbaikan sistem
- *Performance* : Kinerja dalam penggunaan sistem akan tergantung pada komponen perangkat keras dan *processor* yang digunakan oleh *client/user*.
- *Supportability* : Adanya dukungan secara teknis oleh petugas operasional, dalam kaitan melakukan panduan atas adanya permasalahan dalam proses penggunaan Sistem.
- *Portability* : Sistem ini berjalan pada *platform* atau sistem operasi apa saja yang mendukung database phpmyadmin / localhost.
- *Legalitas, Copyright dan Other Notices* : Hak cipta perangkat lunak sistem pencarian rute terpendek menuju rumah sakit menjadi milik pengembang proyek dan pihak instansi. Masing masing pihak tidak dapat mendistribusikan perangkat lunak kepada pihak lain tanpa adanya kesepakatan bersama.

2.15. Tugas Praktikum 2

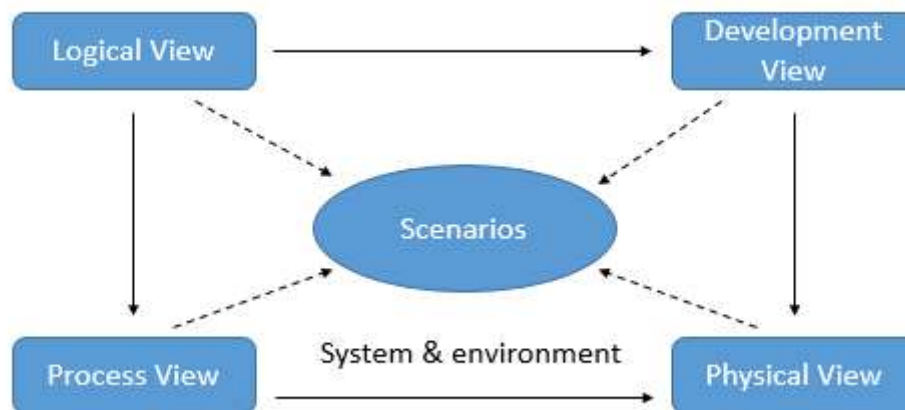
Buatlah overall deskripsion analysis SRS pada kasus tugas praktikum pertemuan 1

MODUL 3

UNIFIED MODELLING LANGUAGE (UML)

3.1. Pengenalan *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah salah satu alat bantu dalam bahasa pemodelan yang digunakan untuk menspesifikasikan, memvisualisasikan, membangun, dan mendokumentasikan suatu sistem yang berorientasi objek. UML adalah hasil kerja dari konsorsium berbagai organisasi yang berhasil dijadikan sebagai standar dalam OOAD (*Object Oriented Analysis & Design*). UML dibangun atas model 4+1 view. Model ini didasarkan pada fakta bahwa struktur sebuah sistem dideskripsikan dalam 5 *view* dimana salah satu diantaranya *scenario*. *Scenario* ini memegang peran khusus untuk mengintegrasikan *content* ke *view* yang lain.



Gambar 3. 1 Model 4+1 View

- **Scenario** : menggambarkan interaksi diantara objek dan di antara proses *scenario* ini digunakan untuk identifikasi elemen arsitekter, ilustrasi, dan validasi desain.
- **Development view** : menjelaskan sebuah sistem dari perspektif programmer dan terkonsentrasikan ke manajemen perangkat lunak.
- **Logical view** : terkait dengan fungsionalitas sistem yang dipersiapkan untuk pengguna akhir.
- **Physical View** : menggambarkan sistem dari perspektif sistem engineer.
- **Process View** : berhubungan erat dengan aspek dinamis dari sistem, proses, yang terjadi di sistem dan bagaimana komunikasi yang terjadi di sistem serta tingkah laku sistem yang dijalankan, dan menjelaskan apa itu *concurrency*, distribusi integrasi, kinerja dan lain-lain.

Bagian-bagian utama dari UML adalah view, diagram, model element, dan general mechanism.

1. View

View digunakan untuk melihat sistem yang dimodelkan dari beberapa aspek yang berbeda. View bukan melihat grafik, tapi merupakan suatu abstraksi yang berisi sejumlah diagram. Beberapa jenis view dalam UML antara lain: use case view, logical view, component view, concurrency view, dan deployment view.

2. Diagram

Diagram berbentuk grafik yang menunjukkan simbol elemen model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu view tertentu dan ketika digambarkan biasanya dialokasikan untuk view tertentu. Adapun jenis diagram antara lain use case diagram, class diagram, state diagram, sequence diagram, collaboration diagram, activity diagram, component diagram, deployment diagram.

3.2. Tipe Diagram UML




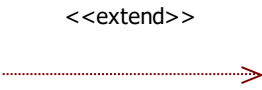
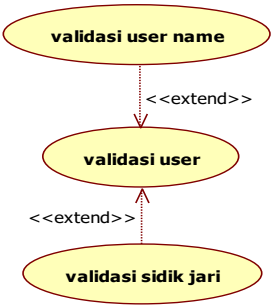

Tabel 3. 1 Tipe Diagram UML

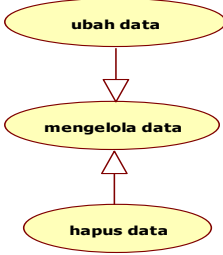
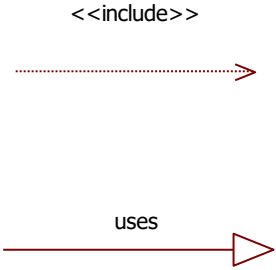
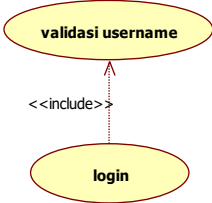
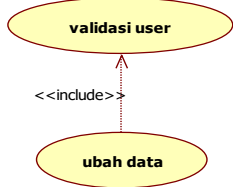
No	Diagram	Tujuan
1	Use Case	Bagaimana user berinteraksi dengan sebuah sistem
2	Activity	Perilaku prosedural & paralel
3	Sequence	Interaksi antar objek, lebih menekankan pada turunan
4	Class	Class, fitur dan relasinya
5	Communication	Interaksi diantara objek. Lebih menekankan ke link
6	Component	Struktur dan koneksi dari komponen
7	Composite Structure	Dekomposisi sebuah class saat runtime
8	Deployment	Penyebaran/instalasi klien
9	Interaction Overation	Gabungan antara activity & sequence diagram
10	Object	Contoh konfigurasi instance
11	Package	Stuktur hierarki saat komplikasi
12	State Machine	Bagaimana event mengubah sebuah objek
13	Timing	Interaksi antar objek, lebih menekankan pada waktu

3.3. Simbol Diagram *Unified Modeling Language* (UML)

3.3.1. Usecase Diagram

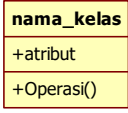






Tabel 3. 2 Usecase Diagram

Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antara unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama use case
Aktor/actor 	Orang, proses, atau sistem lain yang berinteraksi dengan aplikasi yang akan dibuat diluar aplikasi yang akan dibuat itu sendiri, jadi walaupun symbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di frase nama aktor
Asosiasi/association 	Komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan actor
Ekstensi/extend 	<p>Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri sendiri walau tanpa use case tambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya use case tambahan memiliki nama depan yang sama dengan use case yang ditambahkan, misalnya :</p> <div style="text-align: center;">  </div> <p>arah panah mengarah pada use case yang menjadi generalisasinya (umum)</p>
Generalisasi/ Generalization 	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah use case dimana fungsi

	<p>yang satu adalah fungsi yang lebih umum dari lainnya, misalnya :</p>  <pre> graph BT A([hapus data]) -- > B([mengelola data]) B -- > C([ubah data]) </pre> <p>Arah panah mengarah pada use case yang menjadi generalisasinya (umum)</p>
<p>Menggunakan/include/uses</p>  <pre> graph LR subgraph Include A[<<include>>] -.-> B[] end subgraph Uses C[uses] --> D[] end </pre>	<p>Relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya sebagai syarat dijalankan use case ini.</p> <p>Ada dua sudut pandang yang cukup besar mengenai include di use case :</p> <ul style="list-style-type: none"> • include berarti use case yang ditambahkan akan selalu dipanggil saat use case tambahan dijalankan, misalkan pada kasus berikut :  <pre> graph BT A([login]) -.-> <<include>> B([validasi username]) </pre> <ul style="list-style-type: none"> • include berarti use case yang ditambahkan akan selalu melakukan pengecekan apakah use case yang ditambahkan telah dijalankan sebelum use case tambahan dijalankan, misalkan pada kasus berikut :  <pre> graph BT A([ubah data]) -.-> <<include>> B([validasi user]) </pre>


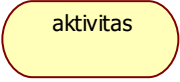
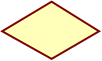
3.3.2. Class Diagram



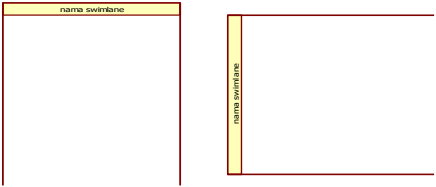
Tabel 3. 3 Class Diagram

Simbol	Deskripsi
<p>Kelas</p> 	Kelas pada struktur sistem
 <p>nama_interface Antarmuka/interface</p>	Sama dengan konsep interface dalam pemrograman berorientasi objek.
<p>Asosiasi/association</p> 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity.
<p>Asosiasi berarah/directed association</p> 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity
<p>Generalisasi</p> 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus)
<p>Menggunakan/include/uses</p> 	Relasi antar kelas dengan makna kebergantungan antar kelas
<p>Agregasi / aggregation</p> 	Relasi antar kelas dengan makna semua-bagian (whole-part)

3.3.3. Activity Diagram


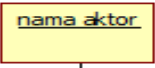

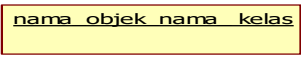

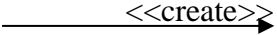
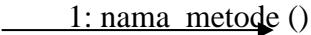
Tabel 3. 4 Activity Diagram




Simbol	Deskripsi
<p>Status awal</p> 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
<p>Aktivitas</p> 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
<p>Percabangan / decision</p> 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu

Penggabungan / joint 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

3.3.4.Sequence Diagram

tabel 3. 5 Sequence Diagram

Simbol	Deskripsi
Aktor  atau  tanpa waktu aktif	Orang, proses, atau sistem lain yang berinteraksi dengan aplikasi yang akan dibuat di luar aplikasi yang akan dibuat itu sendiri, jadi walaupun symbol dari aktor adalah gambar orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor
Garis hidup / line life 	Menyatakan kehidupan suatu objek
Objek 	Nama objek yang berinteraksi pesan
Waktu aktif 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Pesan tipe create 	Menyatukan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe call 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri

Pesan tipe send 	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim
Pesan tipe return 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
Pesan tipe destroy 	Menyatakan bahwa suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy

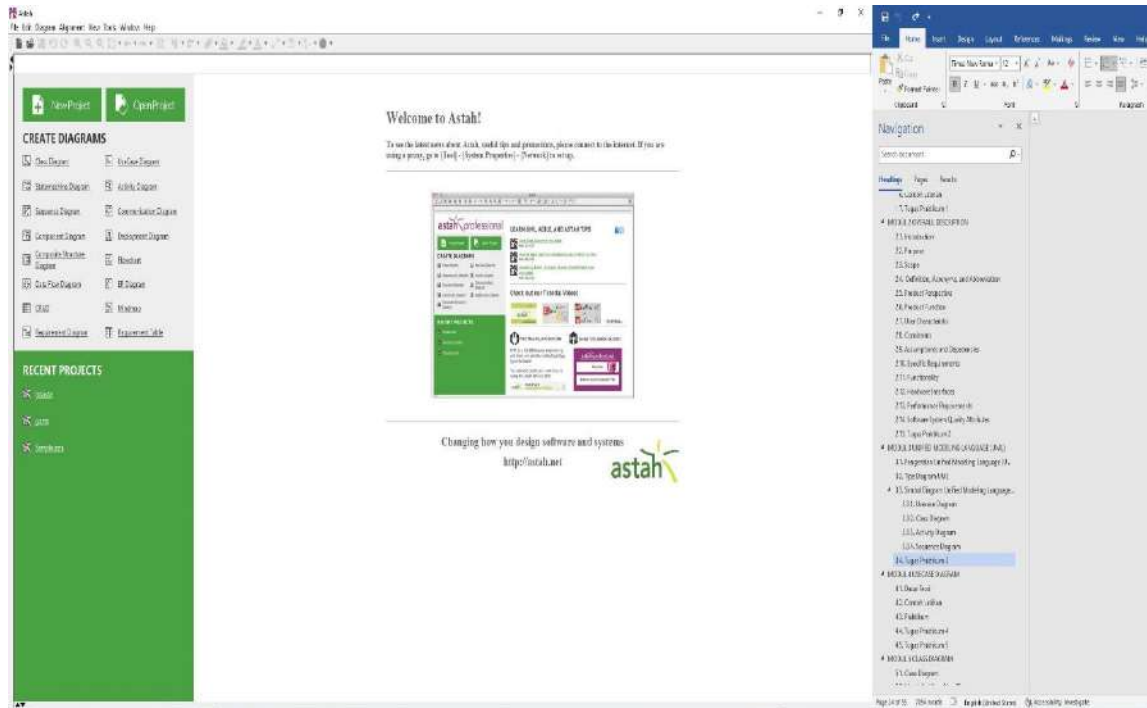
3.4. Astah

Salah satu perangkat lunak untuk membuat model UML adalah Astah. Astah dibuat oleh perusahaan Jepang bernama Change Vision. Astah memiliki produk Astah *Community* dan Astah *Professional* yang masing-masing dapat dioperasikan pada system operasi windows. Astah community merupakan tool gratis dengan fitur-fitur dasar, dilengkapi dengan fitur mencetak diagram, mengimport/mengekspor ke/dari program Java. Sedangkan Astah Professional adalah edisi komersil, semua fitur astah community dapat ditemukan disini, dilengkapi dengan manajemen proyek sebagai fitur yang memungkinkan kolaborasi antar anggota tim proyek. Fitur tambahan lain adalah panduan untuk membuat diagram, model berukuran besar, dan membuat dokumentasi.

3.5. Praktikum

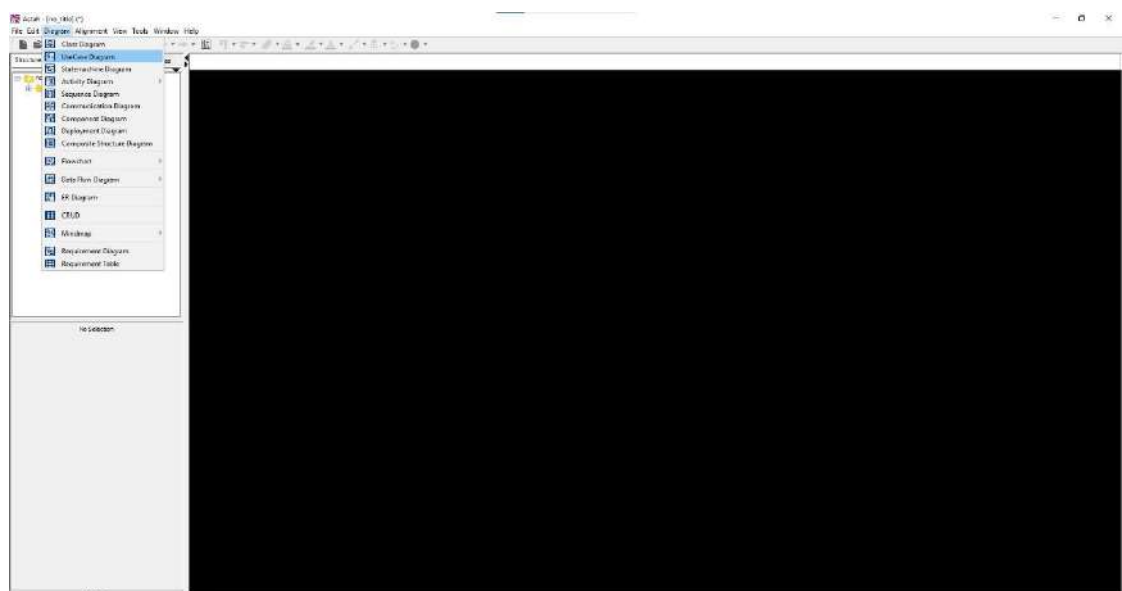
Menjalankan Astah :

1. Buka Aplikasi Astah → Pilih Menu **New Project** Pada Aplikasi astah yang sudah di jalankan.



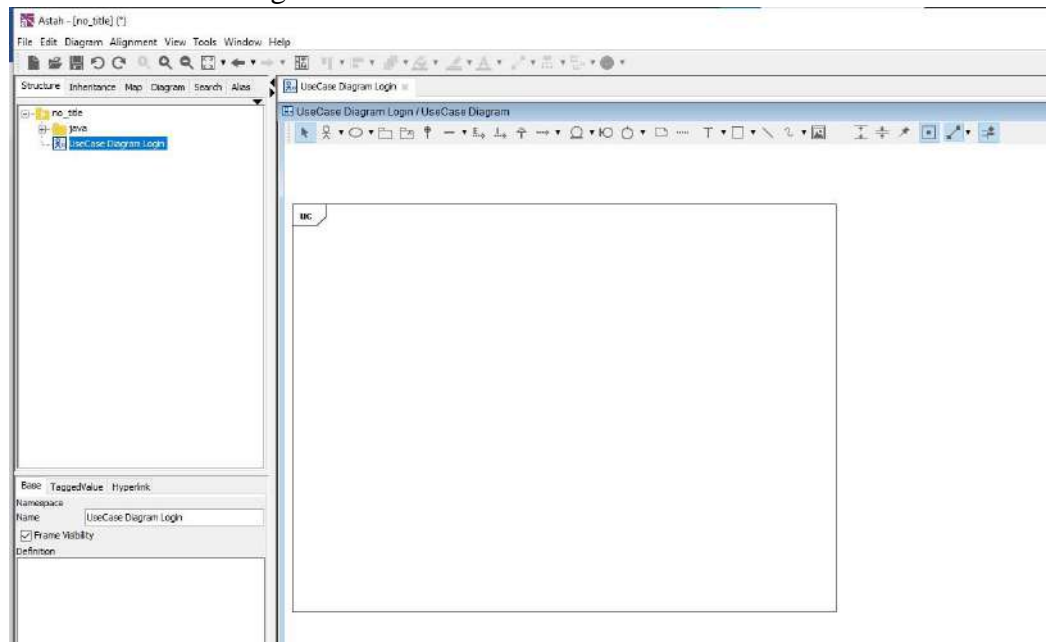
Gambar 3. 2 Astah Professional

2. Pada halaman **New Project** → **Klik menu diagram** pada toolbar diatas dan memilih salah satu model dari UML



Gambar 3. 3 Halaman New Project

3. **Klik menu diagram** pada toolbar diatas dan memilih salah satu model dari UML → **Klik Usecase diagram** (Simbol-simbol diagram usecase akan muncul dibagian atas halaman usecase diagram).



Gambar 3. 4 Jendela Open Usecase Diagram

3.6. Tugas Praktikum 3

Cari contoh suatu kasus dalam penggunaan diagram Usecase, Class, Activity dan sequence Diagram masing-masing satu diagram dan terapkan pada aplikasi astah.

MODUL 4

USECASE DIAGRAM

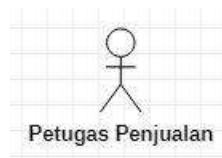
4.1. Dasar Teori

Diagram *use case* menunjukkan beberapa use case dalam sistem, beberapa actor dalam sistem, dan relasi antar mereka dan juga *use case* diagram digunakan untuk menggambarkan sejumlah *external actors* dan hubungannya ke *use case* yang diberikan oleh sistem. Use case adalah deskripsi fungsi yang disediakan oleh system dalam bentuk teks sebagai dokumentasi dari use case symbol namun dapat juga dilakukan dalam activity diagram. Use case digambarkan hanya yang dilihat dari luar oleh actor (keadaan lingkungan sistem yang dilihat user) dan bukan bagaimana fungsi yang ada di dalam sistem.

1. Aktor

Aktor merupakan semua yang ada diluar lingkup sistem perangkat lunak dan berinteraksi dengan sistem perangkat lunak tersebut.

Berikut actor yang dipresentasikan menggunakan notasi stik sebagai berikut



Tipe actor :

- Pengguna sistem
- Sistem lain yang berhubungan dengan sistem yang sedang dibangun
- Waktu

2. Usecase

Usecase menggambarkan bagaimana seseorang sebagai pengguna interaksi dengan sistem. Lebih mudahnya *use case* dapat dikatakan sebagai fungsi-fungsi atau fitur-fitur apa saja yang disediakan oleh sistem informasi yang akan dibangun kepada pengguna, *usecase* bisa juga meliputi apa yang pengguna akan dapat dilakukan terhadap sistem.

Catatan: Use case diagram adalah penggambaran sistem dari sudut pandang pengguna sistem tersebut (user), sehingga pembuatan use case lebih dititik beratkan pada fungsionalitas yang ada pada sistem, bukan berdasarkan alur atau urutan kejadian

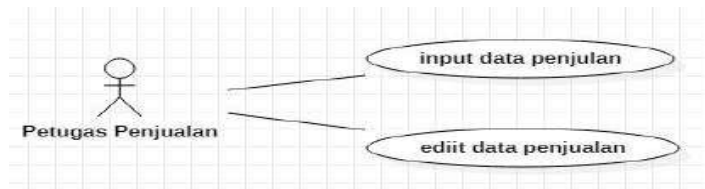


Use Case

Relasi dalam Use Case

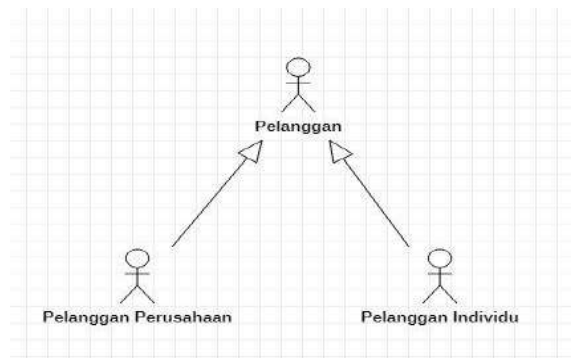
Ada beberapa relasi yang terdapat pada use case diagram:

1. Association, menghubungkan link antar element, untuk menghubungkan antara actor dan use case.



Gambar 4. 1 Relasi asosiasi antar actor dan usecase

2. Generalization, disebut juga inheritance (pewarisan), sebuah elemen dapat merupakan spesialisasi dari elemen lainnya.

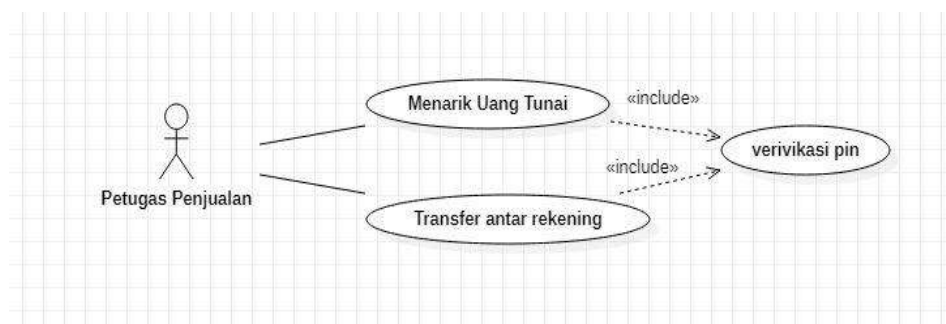


Gambar 4. 2 Relasi Generalisasi

3. Dependency, sebuah element bergantung dalam beberapa cara ke element lainnya
4. Aggregation, bentuk assosiation dimana sebuah elemen berisi elemen lainnya.

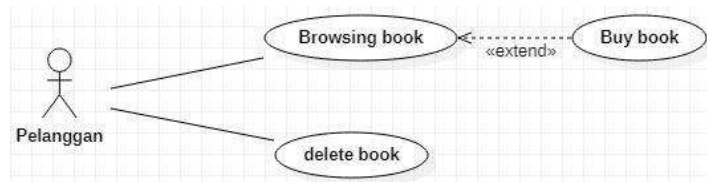
Tipe relasi/ stereotype yang mungkin terjadi pada use case diagram:

- a. <<include>>, yaitu kelakuan yang harus terpenuhi agar sebuah event dapat terjadi, dimana pada kondisi ini sebuah use case adalah bagian dari use case lainnya.



Gambar 4. 3 Relasi *include*

- b. <<extends>>, kelakuan yang hanya berjalan di bawah kondisi tertentu, memungkinkan satu usecase secara opsional bisa dilakukan, bisa tidak dilakukan.



Gambar 4. 4 Relasi Extend

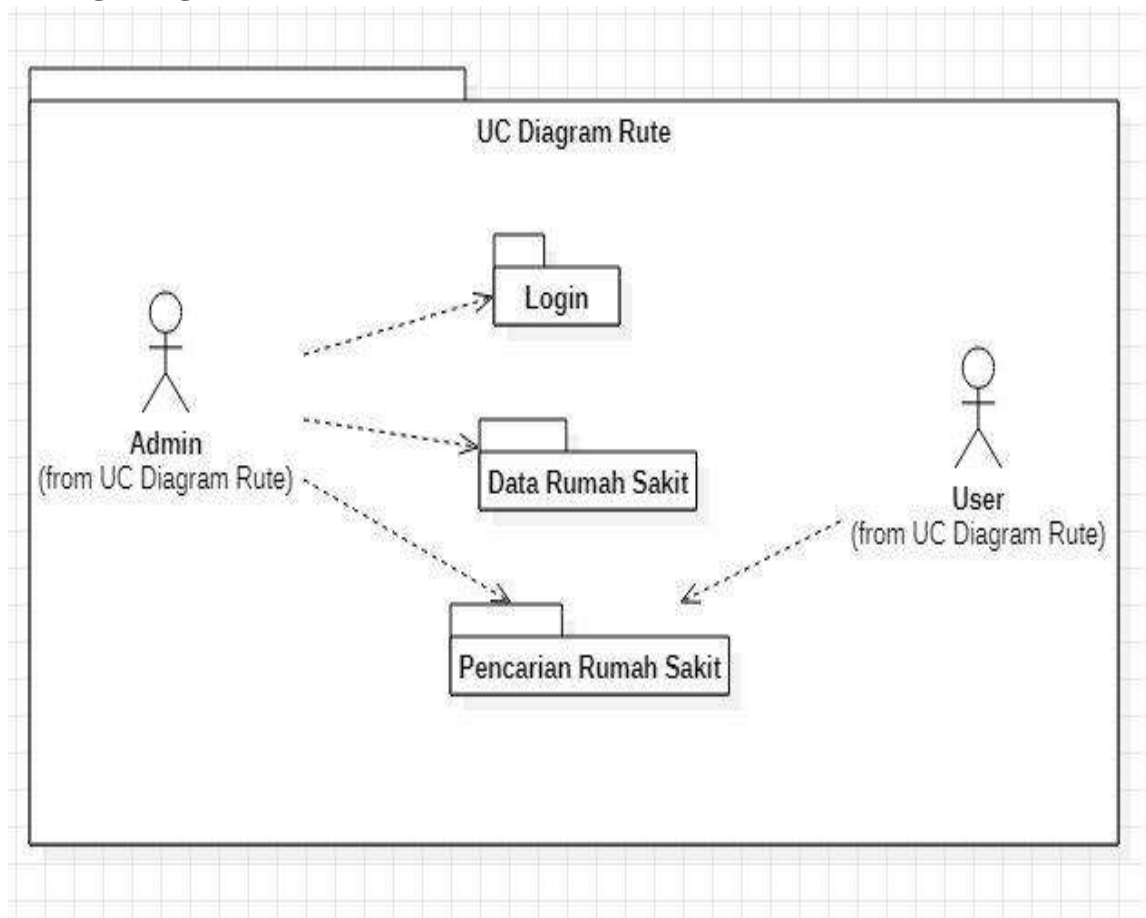
- c. <<communicates>>, mungkin ditambahkan untuk asosiasi yang menunjukkan asosiasinya adalah communicates association. Ini merupakan pilihan selama asosiasi hanya tipe relationship yang dibolehkan antara actor dan use case.

4.2. Contoh Latihan

SKENARIO :

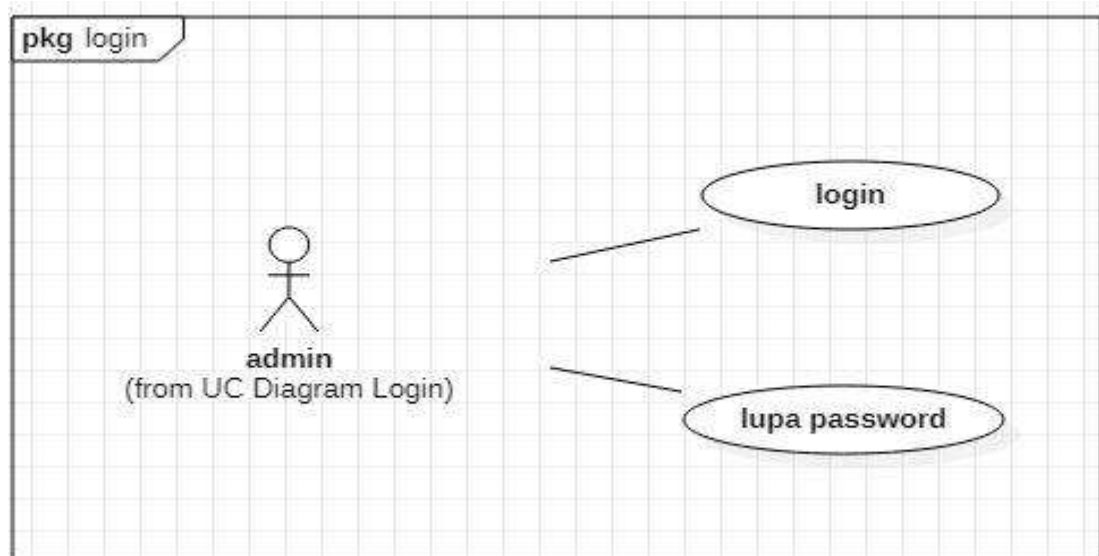
Sebuah pencarian rumah sakit merupakan hal sangat penting ketika ada orang yang hendak pergi ke sebuah rumah sakit dengan mencari jalan/rute terdekat. sistem pencarian rute terpendek ini merupakan perangkat lunak yang digunakan untuk membantu masyarakat mempermudah pencarian rumah sakit dan rute terpendek menuju rumah sakit. Aplikasi ini berkaitan dengan entitas luar yaitu masyarakat umum. Sistem ini merupakan aplikasi informasi pencarian rute terpendek menggunakan algoritma *shortest path greedy*. Ada dua actor yang akan menggunakan sistem ini yaitu admin dan masyarakat umum

1. Package Diagram Usecase



Gambar 4. 5 Package Diagram Usecase

2. Usecase Diagram Login



Gambar 4. 6 2. Usecase Diagram Login

a. Use Case Diskripsi Login

Tabel 4. 1 Usecase deskripsi login

Use Case Name	:	Login	
Triger Event	:	Aktor memilih menu login pada halaman utama	
Aktor	:	admin	
Pre-condition	:	Admin diberikan akun yang detail untuk mengakses ke dalam sistem	
Post-condition	:	admin dapat mengakses ke dalam sistem	
Normal Course		Aktor	Sistem
		1. Mengisi data sesuai kolom yang tersedia	
			2. Validasi kelengkapan data pada form login
			3. Sistem akan menampilkan pesan berhasil login
			4. Menampilkan halaman home setelah login
Alternative Course	:	Aktor	Sistem
			3a. Muncul pesan karena data yang diisi tidak sesuai dengan form username dan password salah
		3b. User dapat memperbaiki data yang salah dan melanjutkan login	

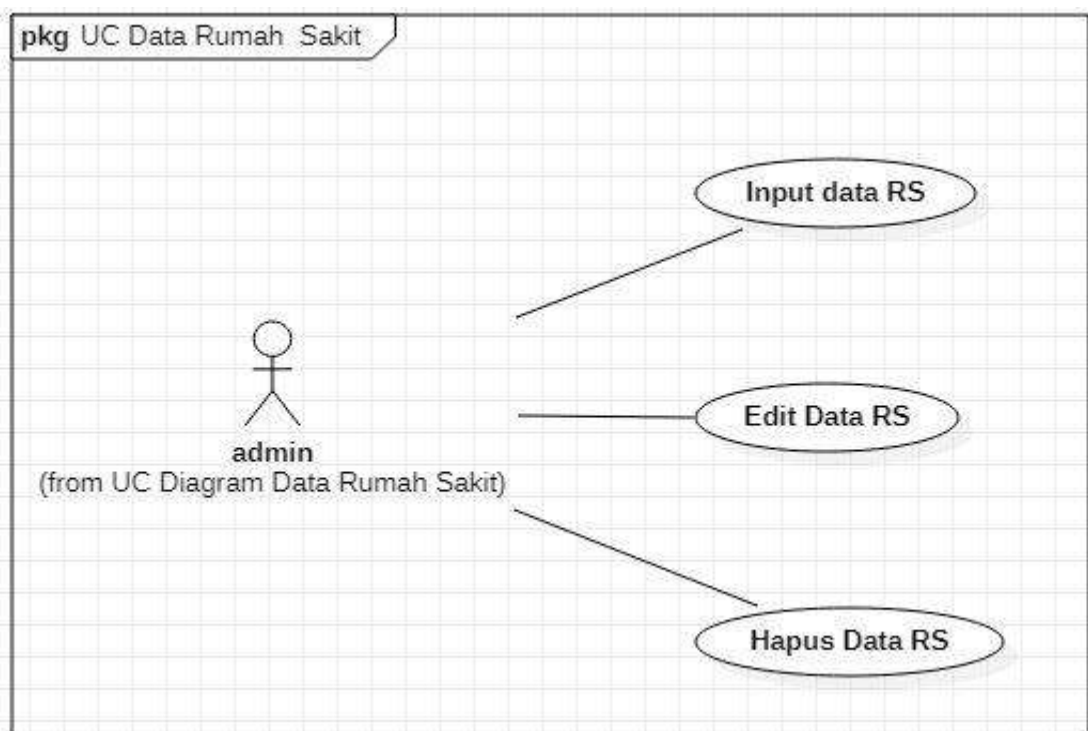
b. Usecase Deskripsi Lupa Password

Tabel 4. 2 Usecase Deskripsi Lupa Password

Use Case Name	:	Lupa password	
Triger Event	:	Aktor memilih menu lupa password pada halaman utama	
Aktor	:	Admin	
Pre-condition	:	Admin memasukan user name, password dan kode sistem	
Post-condition	:	User dapat mengakses ke dalam sistem	
Normal Course		Aktor	Sistem
		1. Mengisi data sesuai kolom yang tersedia	
			2. Validasi kelengkapan data pada form lupa password
			3. Sistem akan menampilkan pesan berhasil password di rubah

			4. Menampilkan halaman login
Alternative Course	:	Aktor	Sistem
			3c. Muncul pesan karena data yang diisi tidak sesuai dengan form
		3d. User dapat memperbaiki data yang salah dan melanjutkan pengisian form lupa password	
Exception	:	User dapat membatalkan lupa password dengan klik button batal	

3. Usecase Diagram data Rumah Sakit



Gambar 4. 7 Usecase Diagram data Rumah Sakit

a. Usecase deskripsi input data RS

Tabel 4. 3 Usecase deskripsi input data RS

Use Case Name	:	Input data RS	
Triger Event	:	Aktor memilih menu data rumah sakit pada halaman home	
Aktor	:	Admin	
Pre-conditon	:	<ul style="list-style-type: none"> - Aktor telah login - Aktor telah membuka halaman data rumah sakit 	
Post-condition	:	Aktor telah menambah data rumah sakit dan telah tersimpan di database	
Normal Course	:	Aktor	Sistem
		1. Memilih menu data rumah sakit	
			2. Menampilkan form data rumah sakit
		3. Mengisi form data rumah sakit	
		4. Memilih tombol add	
			5. Melakukan validasi data
			6. Jika data berhasil divalidasi maka tampil pesan data berhasil ditambahkan
		Aktor	Sistem
Alternative Course			6a. Jika data gagal divalidasi maka tampil pesan data gagal ditambahkan
		7. Aktor dapat membatalkan tambah data data rs dengan mengklik tombol exit pilih menu keluar	
Exception	:	Aktor dapat membatalkan tambah data rumah sakit	

b. Usecase deskripsi Edit data RS

Tabel 4. 4 Usecase deskripsi Edit data RS

Use Case Name	:	Edit data RS	
Triger Event	:	Aktor memilih menu data rumah sakit pada halaman home	
Aktor	:	Admin	
Pre-conditon	:	<ul style="list-style-type: none"> - Aktor telah login - Aktor telah membuka halaman data rumah sakit 	
Post-condition	:	Aktor telah mengedit data rumah sakit dan telah tersimpan di database	

Normal Course	:	Aktor	Sistem
		1. Memilih menu data rumah sakit	
			2. Menampilkan data rumah sakit
		3. Memasukan/mencari data rumah sakit yang akan di edit	
			4. Menampilkan data rumah sakit sesuai pencarian
		5. Mengubah form data rumah sakit yang akan di edit	
		6. Klik tombol edit	
			7. Validasi data
			8. Jika validasi berhasil maka tampil pesan data berhasil diedit
		Aktor	Sistem
			8a. Jika validasi berhasil maka tampil pesan data berhasil diedit
Alternative Course	:	Aktor dapat membatalkan edit data rumah sakit dengan mengklik tombol exit pilih menu keluar	
Exception	:	Aktor dapat membatalkan edit data rumah sakit	

c. Use case deskripsi hapus data rumah sakit

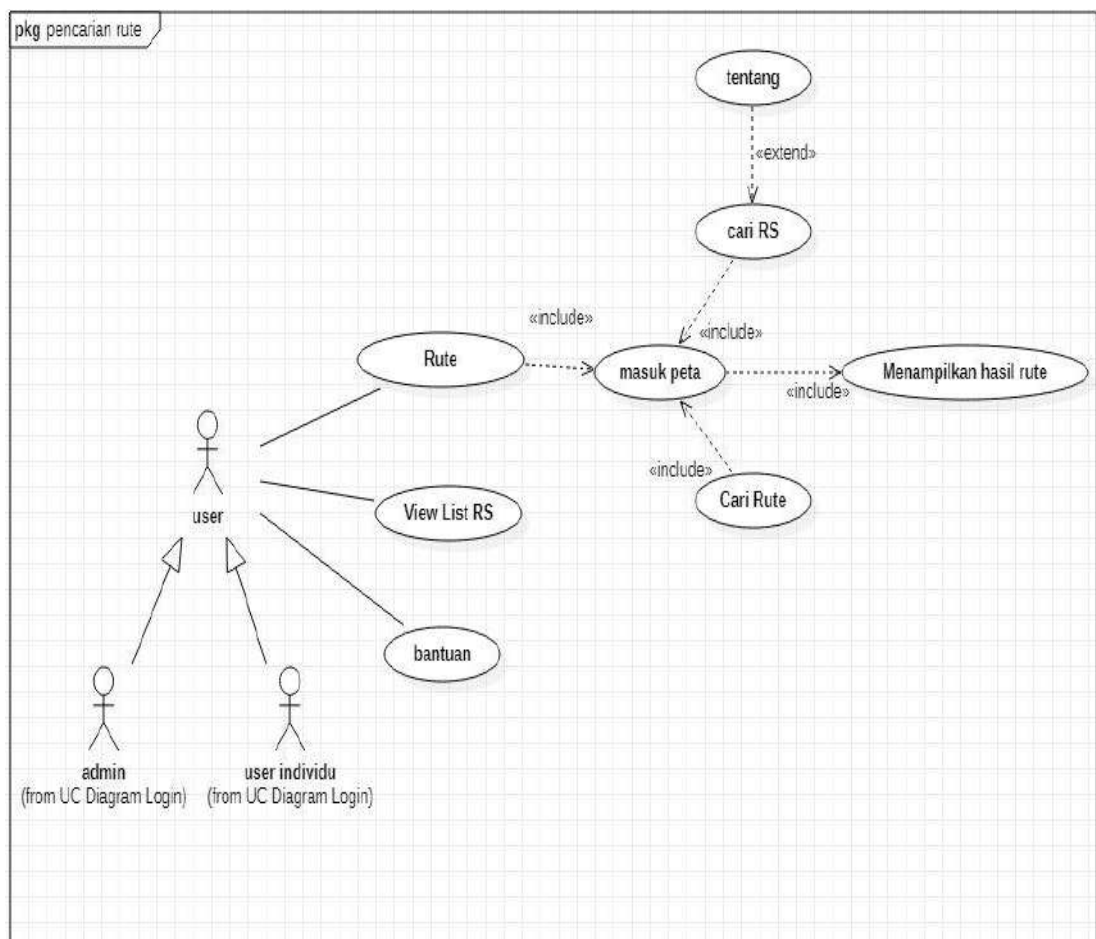
Tabel 4. 5 Use case diagram hapus data rumah sakit

Use Case Name	:	Hapus data RS	
Triger Event	:	Aktor memilih menu data rumah sakit pada halaman home	
Aktor	:	Admin	
Pre-conditon	:	<ul style="list-style-type: none"> - Aktor telah login - Aktor telah membuka halaman data rumah sakit 	
Post-condition	:	Aktor telah menghapus data rumah sakit	
Normal Course		Aktor	Sistem
		1. Memilih menu data rumah sakit	
			2. Menampilkan halaman data rumah sakit
		3. Memasukan/mencari data rumah sakit yang akan di hapus	
			4. Menampilkan data rumah sakit sesuai pencarian

		5. Klik tombol hapus	
			6. Validasi data
			7. Jika validasi berhasil maka tampil pesan data berhasil dihapus
Alternative Course	:	Aktor	Sistem
			7a. Jika validasi gagal maka tampil pesan data gagal dihapus
Exception	:	Aktor dapat membatalkan hapus data rumah sakit dengan klik tombol exit atau pilih menu keluar	

4. Usecase diagram Pencarian rute

Gambar 4. 8 Usecase diagram Pencarian rute



4.3. Praktikum

Buat Usecase deskripsi dari usecase pencarian Rute

4.4. Tugas Praktikum 4

SKENARIO :

Pada studi kasus pemrograman web kita akan membuat sistem Penerimaan Peserta Didik Baru (PPDB) di sekolah. Berikut prosedur yang berjalan pada pelaksanaan PPDB.

Analisis Kebutuhan

- a. Pendaftaran
 - Calon peserta didik mengisi formulir pendaftaran online atau offline dengan data pribadi dan informasi pendidikan.
 - Data yang biasanya diminta mencakup nama lengkap, alamat, tanggal lahir, riwayat pendidikan, dan kontak.
 - Dalam formulir, calon peserta didik mungkin juga harus memilih program studi atau jurusan yang diminati.
- b. Seleksi dan Penilaian
 - Sekolah dapat melakukan seleksi berdasarkan kriteria tertentu, seperti nilai rapor, tes tertulis, wawancara, atau tes keterampilan.
 - Proses ini dapat melibatkan tim penerimaan dan guru-guru terkait.
- c. Pengumuman Hasil Seleksi
 - Hasil seleksi diumumkan secara transparan dan akurat. Calon peserta didik yang lulus seleksi diberitahu melalui email, telepon, atau situs web sekolah.

1. Buat usecase diagram pada scenario diatas
2. Buat usecase deskripsi pada scenario diatas.

4.5. Tugas Praktikum 5

SKENARIO :

Sistem ATM yang akan dibuat modelnya adalah system ATM sederhana dan umum dengan fungsi mendasar yaitu, mengambil uang tunai, melakukan transfer ke rekening tertentu pada bank yang sama, dan memeriksa saldo.

System yang dimodelkan tidak mencakup beberapa fungsi ATM yang tergantung fasilitas tambahan dari bank misalnya membayar kartu kredit, membayar telepon, membayar tagihan seluler, mengisi pulsa, membayar PBB, ATM bersama yang menerima berbagai kartu, dan lain-lain, karena layanan-layanan tersebut berbeda-beda tergantung dari operator/bank yang mengeluarkan ATM tersebut.

Buat usecase diagram dan usecase deskripsi pada kasus di atas.

MODUL 5

CLASS DIAGRAM

5.1. Class Diagram

Diagram kelas menggambarkan jenis-jenis objek dalam suatu system dan berbagai jenis hubungan statis yang ada diantaranya. Sebuah kelas merupakan kumpulan dari objek yang memiliki karakteristik yang sama seperti atribut, operasi hubungan dan semantic. Sebuah kelas mengimplentasikan satu atau lebih *interface*. (Hamim Tohari 2015)

5.2. Mendefinisikan Class Diagram

Kelas (*class*) adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan perancangan berorientasi objek. Kelas menggambarkan keadaan (*atribut/property*) suatu system, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (*metode/fungsi*).

Dalam pemodelan statis dari sebuah system, diagram kelas biasanya digunakan untuk memodelkan salah satu dari tiga hal berikut :

1. Perbendaharaan dari system
2. Kolaborasi
3. Skema basis data *logical*

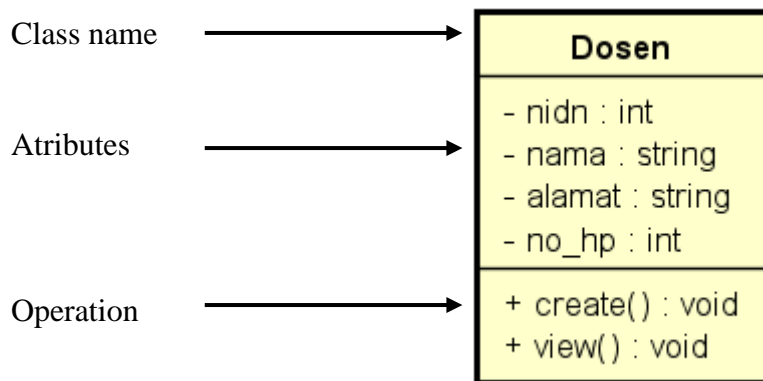
Kelas memiliki tiga area pokok :

1. Nama/ Entitas (dan *stereotype*)
2. Atribut
3. Metode dan operasi

Atribut dan metode dapat memiliki salah satu sifat berikut :

1. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan
2. *Protected*, hanya dapat dipanggil dari luar *class* yang bersangkutan dan anak-anak yang mewarisinya
3. *Public*, dapat dipanggil oleh siapa saja

Kelas dapat berupa implemtasi dari sebuah *interface*, yaitu *class* abstrak yang hanya memiliki metode. *Interface* tidak dapat langsung diinstansikan, tetapi harus diimplemntasikan dahulu menjadi sebuah *class*. Dengan demikian *interface* mendukung resolusi metode pada saat *run-time*.



5.3. Cara mengidentifikasi kelas

Untuk mengidentifikasi kelas dari suatu system dapat dimulai dari model *usecase* yang sudah dibuat dengan cara menetapkan objek apa saja yang dibutuhkan di system. Objek dapat berupa orang, benda, kejadian, tempat dan lain sebagainya. Setiap objek memiliki atribut-atribut dan metode/operasinya.

Berikut ini beberapa alternatif cara mendefinisikan kandidat kelas :

1. Mengidentifikasi kelas-kelas dan objek-objek yang ada di dalam lingkup system :
 - a. Dari uraian pernyataan masalah
 - b. Secara menyeluruh pada lingkup kebutuhan system atau pengetahuan atas lingkup aplikasi
 - c. Dari kebutuhan fungsional system dan atau kebutuhan pemakai
2. Kelas dan objek dapat di identifikasikan dari salah satu atau lebih hal berikut :
 - a. Entitas eksternal yang memproduksi dan memakai informasi yang akan digunakan oleh system berbasis computer.
 - b. Sesuatu yang merupakan bagian dari wilayah informasi dari permasalahan.
 - c. Kejadian, misalnya prosedur operasional, yang muncul dalam lingkup operasional system.
 - d. Peran yang dimainkan oleh orang-orang yang berintersksi dengan system.
 - e. Unit organisasi yang relavan dengan aplikasi dan perlu dikelola datanya.
 - f. Tempat yang menentukan ruang lingkup masalah dan seluruh fungsi system
 - g. Struktur yang mendefinisikan kelas dari objek atau yag menghubungkan kelas-kelas objek.
3. Baikan kelas dan objek yang tidak tepat karena :

Redudan, tidal relavan, samar, lebih tepat berupa atribut, lebih tepat berupa operasi, lebih tepat berupa peran, atau lebih merupakan konstruksi implementasi.

4. Setelah kelas terdefinisi, identifikasi atribut dan metode setiap kelas. Atribut diidentifikasi dari elemen-elemen data yang dapat menggambarkan ciri-ciri sebuah objek secara utuh. Metode atau operasi atau layanan diidentifikasi dari perilaku, khususnya yang dapat menunjukkan peran dan tanggung jawab suatu objek.
5. Menefinisikan hubungan (asosiasi atau koneksi) antar kelas, yaitu ketergantungan antar satu kelas atau lebih dengan kelas lainnya.

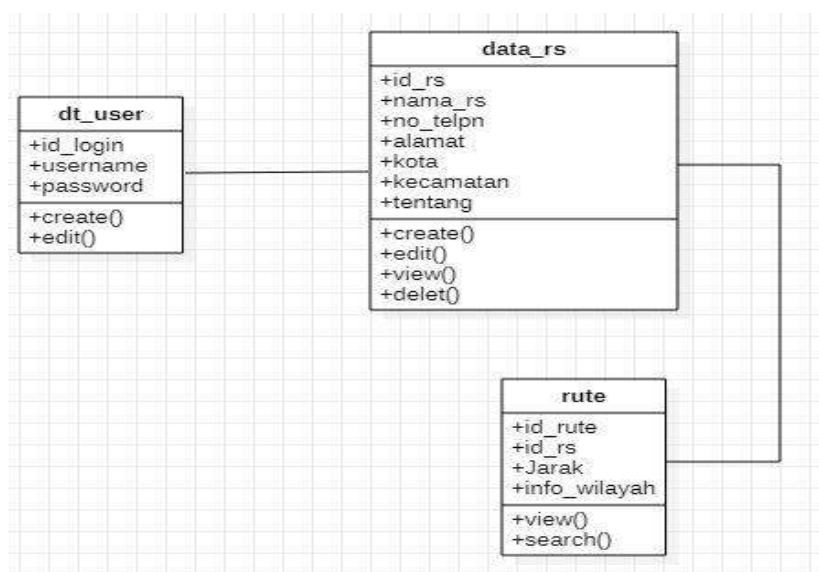
5.4. Nilai Kardinalitas

Nilai kardinalitas atau *multiplicity* atau multiplisitas menunjukkan jumlah suatu objek yang dapat berhubungan dengan objek lain. Multiplisitas biasanya ditunjukan dengan “satu” atau “banyak”, tetapi secara khusus dapat ditunjukan pula dengan bilangan integer lebih besar atau sama dengan nol.

Tabel 5. 1 jenis-jenis Multiplicity

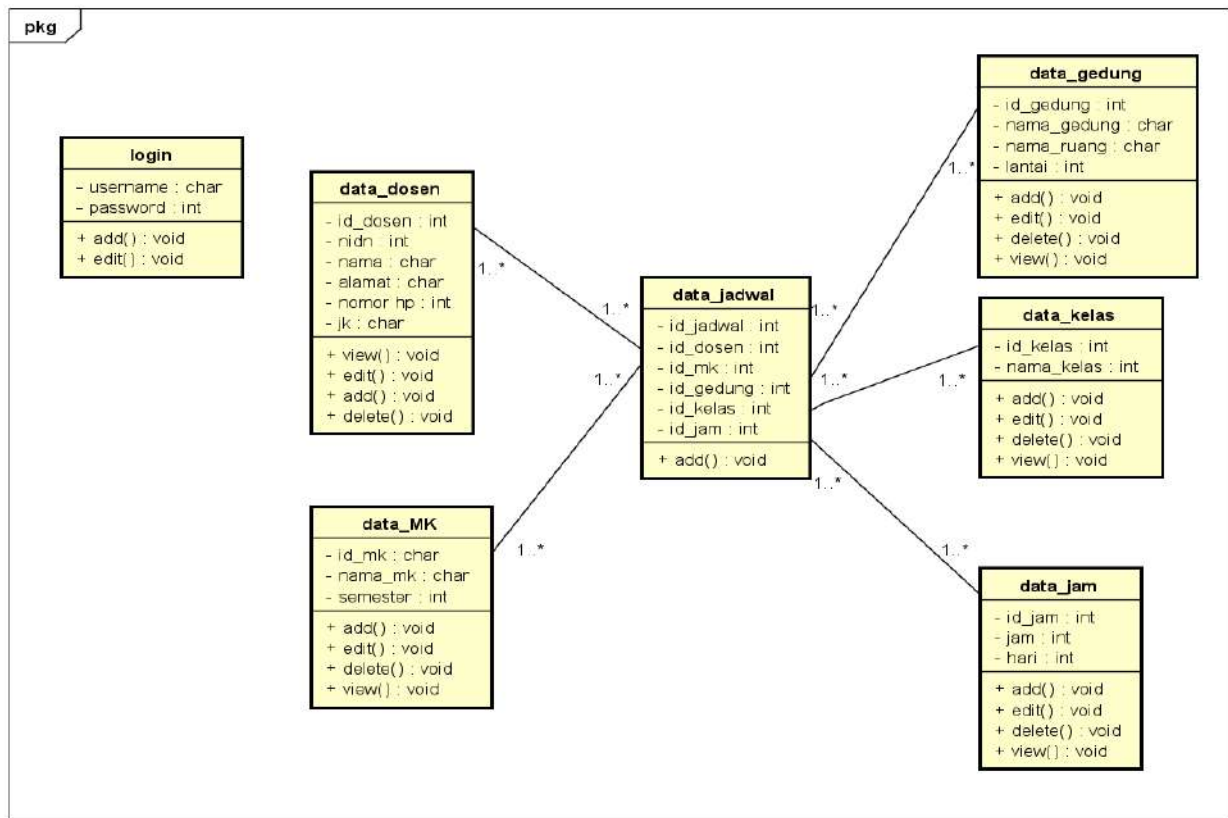
Indikator	Arti
0..1	Nol atau Satu
1	Hanya satu
0.. *	Nol atau lebih
1.. *	Satu atau lebh
n	Hanya n (dengan $n > 1$)
0.. n	Nol sampai n (dengann > 1)
1. n	Satu sampai n (dengann > 1)

5.5. Contoh Class Diagram



Gambar 5. 1 Diagram Class

5.6. Latihan Praktikum



5.7. Tugas Praktikum 6

Buatlah class diagram dari sistem Penerimaan Peserta Didik Baru (PPDB) kasus tugas praktikum 4

5.8. Tugas Praktikum 7

Buatlah Class diagram dari sistem ATM dari kasus tugas praktikum 5

5.9. Uji Pemahaman Materi

Definisikan sistem dibawah ini.

Sistem Perpustakaan yang memiliki kebutuhan sebagai berikut :

- Perpustakaan memiliki koleksi buku, jurnal dan majalah
- Petugas perpustakaan dapat melayani peminjaman, pengembalian, dan pemesanan buku yang sedang dipinjam dan pencarian.
- Anggota perpustakaan (dosan dan mahasiswa) dapat mencari koleksi buku yang akan dipinjam
- Setiap peminjam harus terdaftar menjadi anggota.

Buatlah Analisis, Overall Deskripsi, Usecase Diagram, Use Deskripsi dan class diagram dari kasus di atas

MODUL 6

ACTIVITY DIAGRAM

6.1. Pengertian Activity Diagram

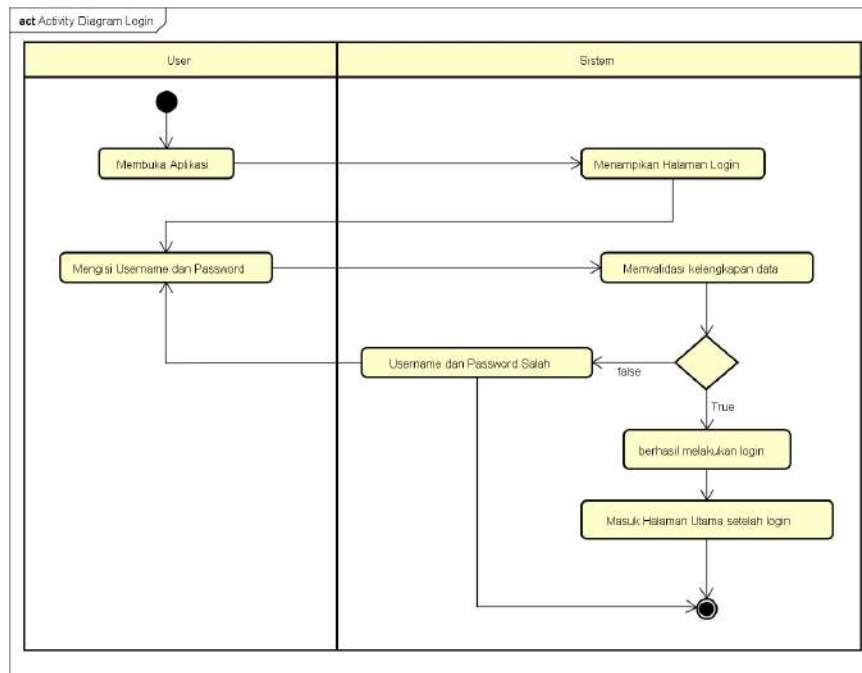
Activity Diagram memodelkan *workflow* proses bisnis dan urutan aktivitas dalam sebuah proses. Diagram ini sangat mirip dengan *flowchart* karena memodelkan *workflow* dari suatu aktivitas lainnya atau aktivitas ke status. Membuat *activity diagram* pada awalnya pemodelan proses cukup menguntungkan untuk membantu memahami keseluruhan proses. Activity diagram juga bermanfaat untuk menggambarkan *parallel behavior* atau menggambarkan interaksi antara beberapa usecase.

6.2. Elemen-Element dari Actyvity Diagram

1. Status *start* (mulai) dan *end* (akhir)
2. Aktivitas yang mempresentasikan sebuah langkah dalam *workflow*
3. *Transition* menunjukkan terjadinya perubahan status aktivitas (*transition show what state follows another*)
4. Keputusan yang menunjukan alternative dalam *workflow*.
5. *Synchoronization bars* yang menunjukkan *subflow parallel*. *Synchoronization bars* dapat digunakan untuk menunjukan *concurrent therads* pada *workflow* proses bisnis.
6. *Swimlanes* yang mempresentasikan *role* bisnis yang bertanggung jawab pada aktivitas yang berjalan.

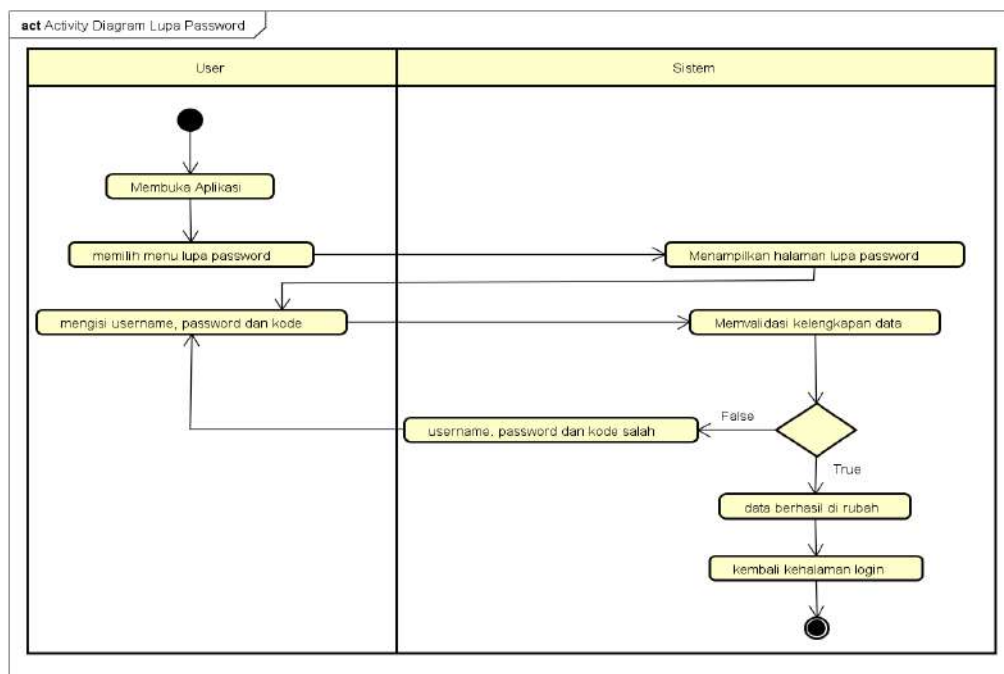
6.3. Contoh Activity Diagram

1. Activity diagram Login



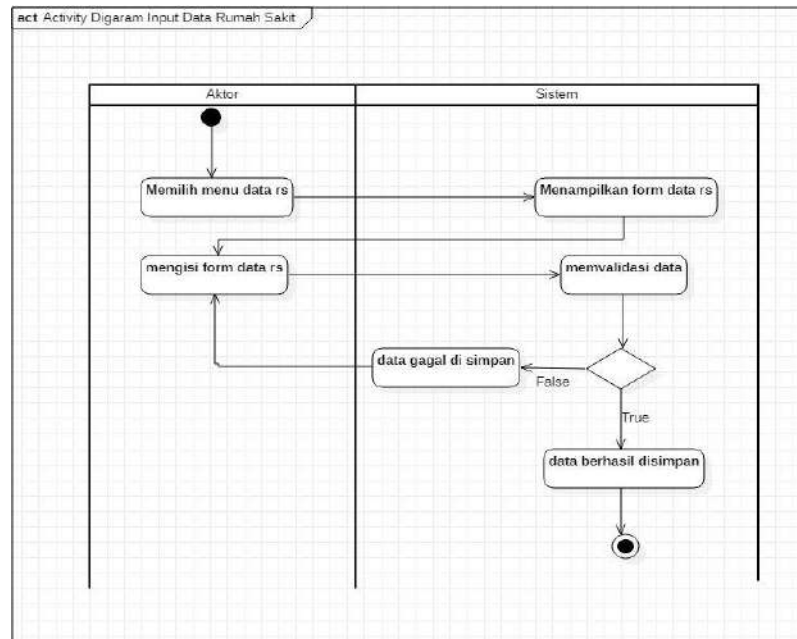
Gambar 6. 1 activity diagram login

2. Activity Diagram Lupa Password



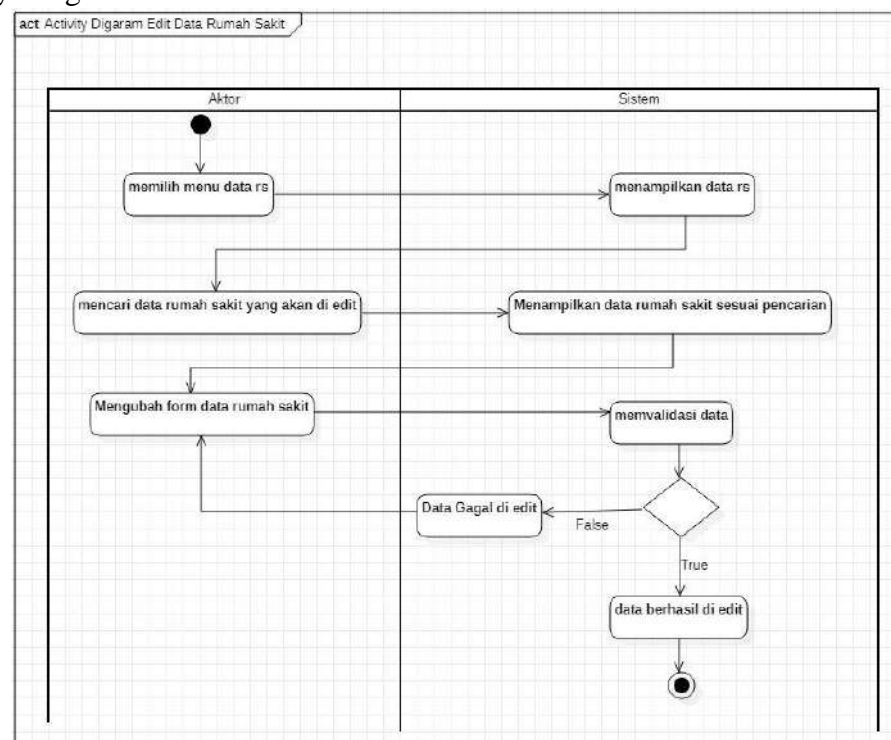
Gambar 6. 2 activity diagram lupa password

3. Activity Diagram Input Data Rumah Sakit



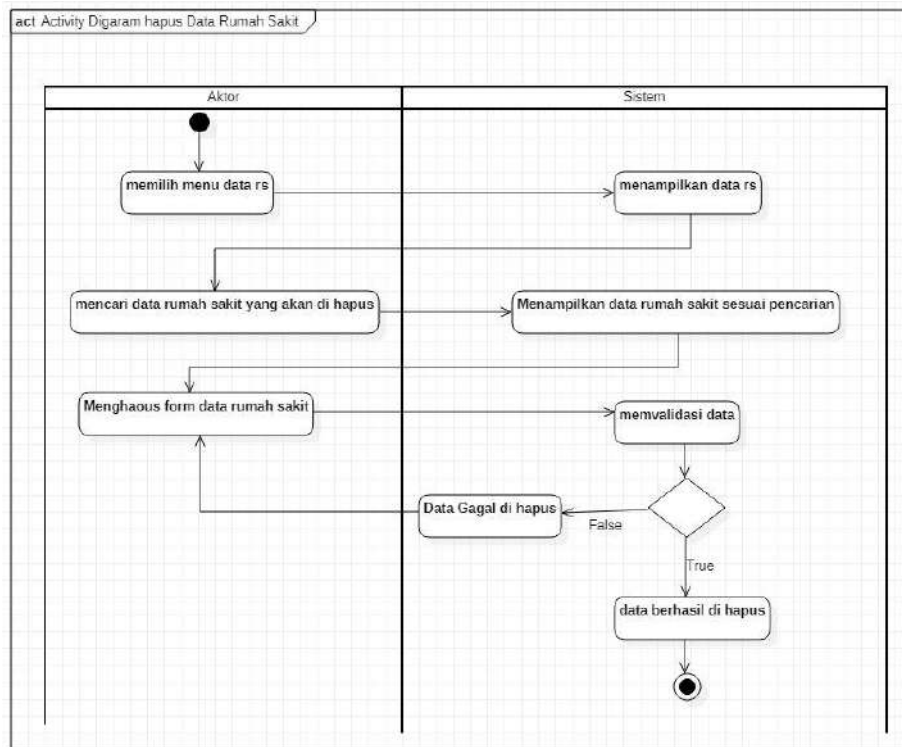
Gambar 6. 3 Activity Diagram Input Data Rumah Sakit

4. Activity Diagram Edit Data Rumah Sakit



Gambar 6. 4 Activity Diagram Edit Data Rumah Sakit

5. Activity Diagram Hapus Data Rumah Sakit



Gambar 6. 5 Activity Diagram Hapus Data Rumah Sakit

6.4. Latihan Praktikum

Buatlah activity diagram dari contoh kasus pencarian rute

6.5. Tugas Praktikum 8

Buatlah Activity diagram sistem informasi Penerimaan Peserta Didik Baru (PPDB) sebelumnya

6.6. Tugas Praktikum 9

Buatlah acivity diagram sistem ATM dari kasus sebelum nya

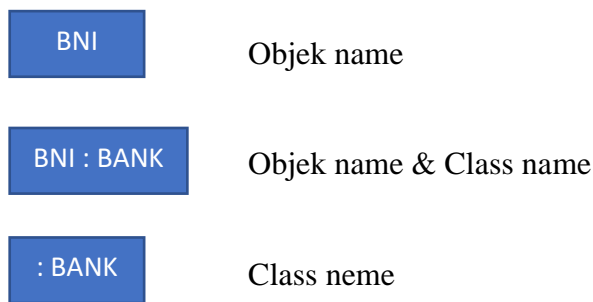
MODUL 7

SEQUENCE DIAGRAM

7.1. Sequence Diagram

Sequence diagram menggambarkan interaksi antara sejumlah objek dalam urutan waktu dan perilaku pada sebuah *scenario*. Diagram ini menunjukkan sejumlah contoh objek dan *message* (pesan) yang diletakan diantara objek-objek ini dalam *usecase*.

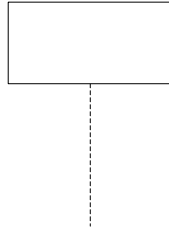
Komponen utama dalam *sequence diagram* terdiri atas objek yang dituliskan dengan kotak segiempat bernama. Message diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan progress vertical, terdapat tiga cara untuk menemani objek yaitu : nama objek, nama objek dan *class* serta nama *class*(Aries Hadi Sutopo 2002)



Pada diagram sequence, setiap objek hanya memiliki garis yang digambarkan garis putus-putus kebawah. Pesan antar objek digambarkan dengan anak panah dari objek yang mengirimkan pesan ke objek yang menerima pesan.

7.2. Obyek/ Participant

Obyek diletakan didekat bagian atas diagram denan urutan dari kiri ke kanan. Sehingga diatur dalam urutan guna menyederhanakan diagram. Setiap obyek terhubung dengan garis titik-titik yang disebut ***lifeline***. Sepanjang *lifeline* ada kotak yang disebut dengan ***activation***. *Activation* mewakili sebuah eksekusi operasi dari obyek. Panjang kotak ini berbanding lurus dengan durasi activation.



Gambar 7. 1 Objek pada sebuah sequence diagram

7.3. *Message*

Sebuah message bergerak dari satu objek ke objek lain dan dari satu *lifeline* ke *lifeline* yang lain. Sebuah objek dapat mengirim sebuah message kepada dirinya sendiri.

7.4. *Time*

Time adalah diagram yang mewakili waktu pada arah vertical. Waktu dimulai dari atas kebawah. *Message* yang lebih dekat dari atas akan dijalankan terlebih dahulu dibandingkan *message* yang lebih dekat kebawah. (Munawar 2018)

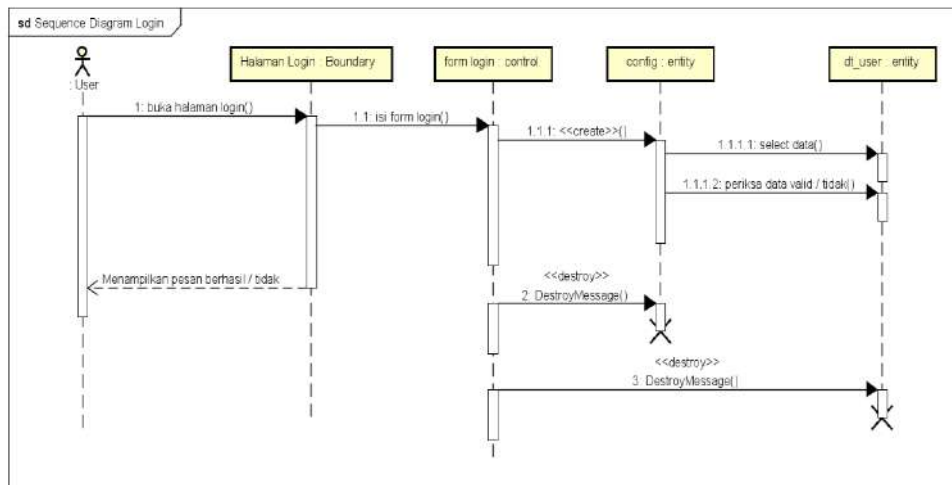
7.5. Tujuan Sequence Diagram

Secara umum sequence diagram menunjukkan urutan waktu aliran pesan dari satu objek ke objek lainnya, namun secara lebih spesifik tujuan sequence diagram bisa digambarkan sebagai berikut :

1. Model interaksi tingkat tinggi antara objek aktif dalam suatu system,
2. Model interaksi antara *intence* (contoh) objek dalam kolaborasi yang merealisasikan *usecase*
3. Model interaksi antar objek dalam kolaborasi yang mewujudkan operasi
4. Mewujudkan model interaksi generic (menunjukkan semua jalur yang mungkin melalui interaksi) atau contoh spesifik dari suatu interaksi (menunjukkan hanya satu jalur melalui interaksi)

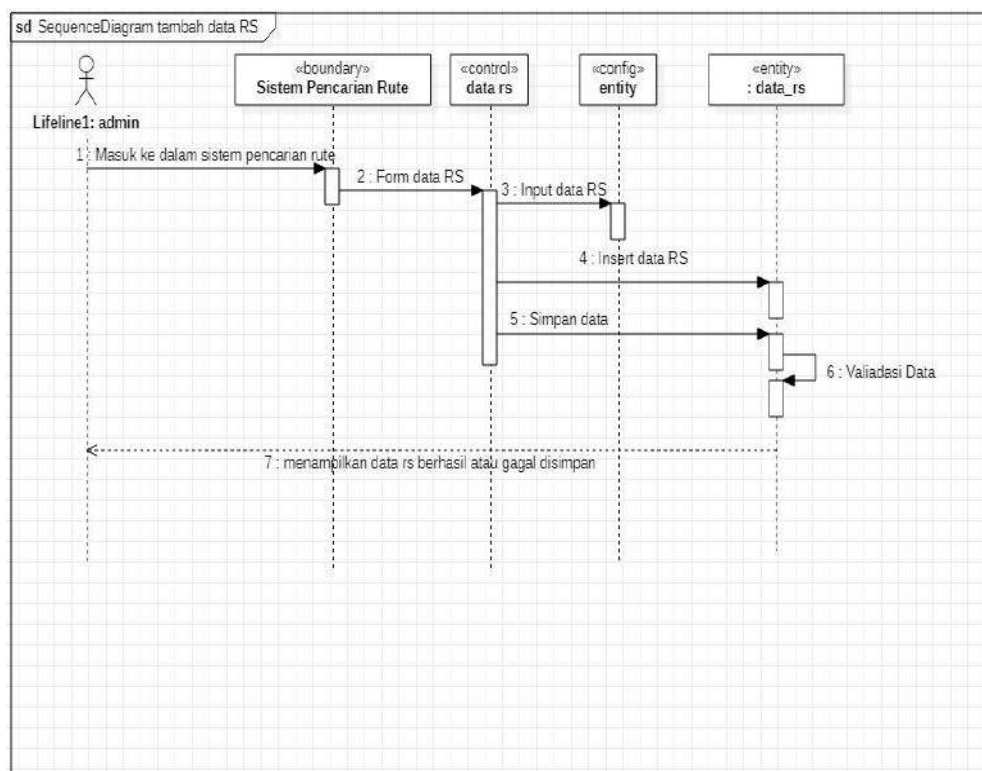
7.6. Contoh Sequence Diagram

1. Sequence Diagram Login



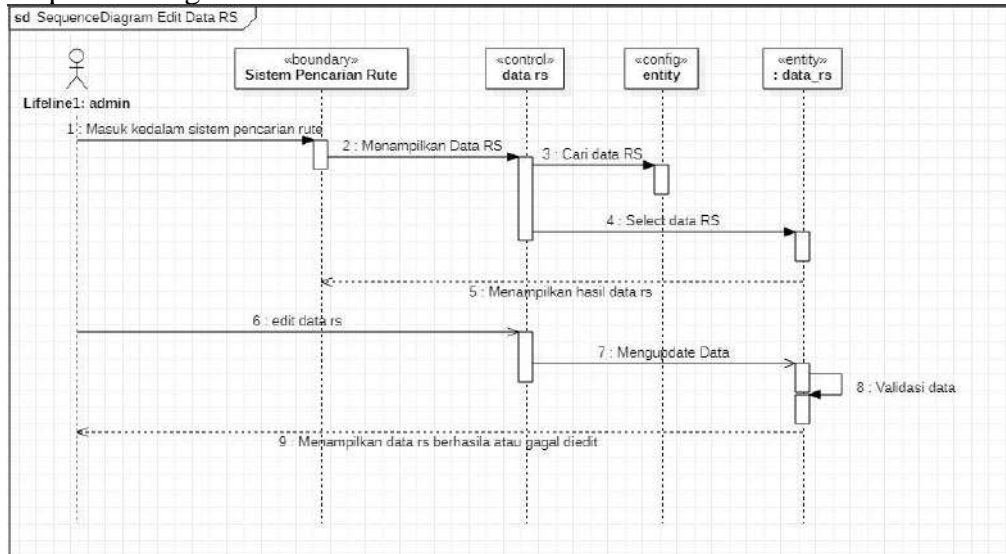
Gambar 7. 2 Sequence Diagram Login

2. Sequence Diagram Input Data RS



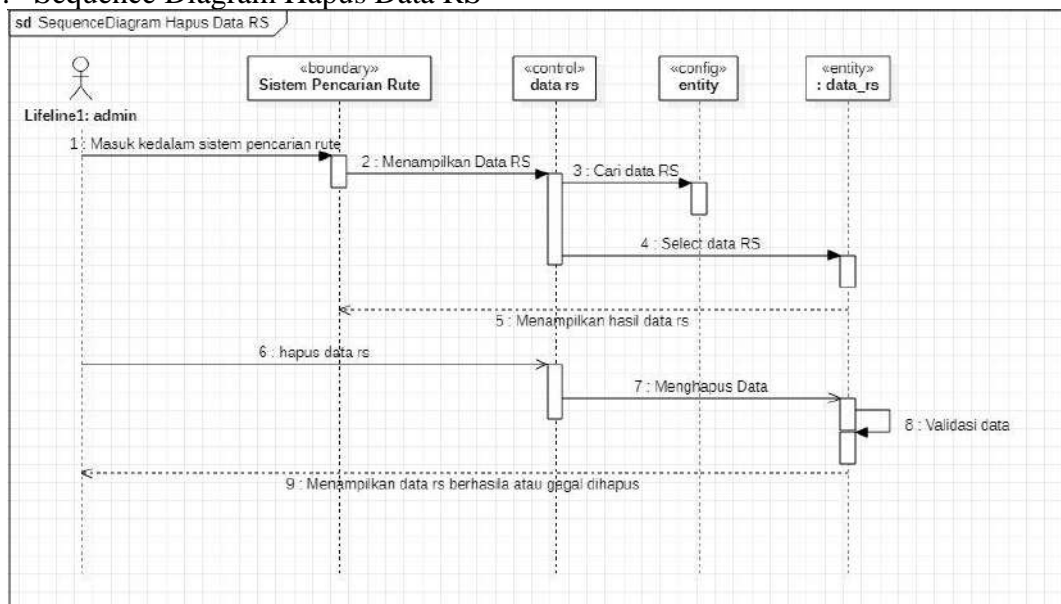
Gambar 7. 3 Sequence Diagram Inpit Data RS

3. Sequence Diagram Edit Data RS



Gambar 7. 4 Sequence Diagram Edit Data RS

4. Sequence Diagram Hapus Data RS



Gambar 7. 5 Sequence Diagram Hapus Data RS

7.7. Latihan Praktikum

Buatlah activity diagram dari contoh kasus pencarian rute

7.8. Tugas Praktikum 8

Buatlah Sequence diagram sistem informasi Penerimaan Peserta Didik Baru (PPDB) sebelumnya

7.9. Tugas Praktikum 9

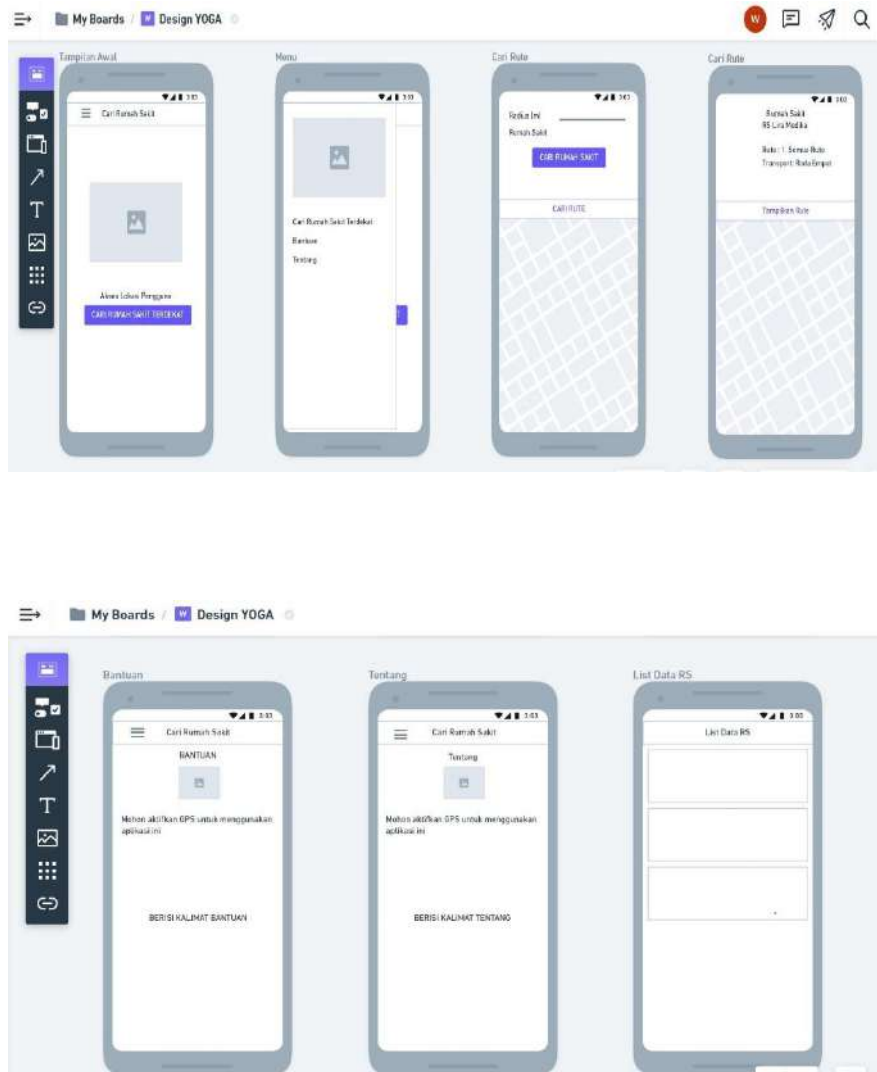
Buatlah Sequence diagram sistem ATM dari kasus sebelum nya.

MODUL 8

USER INTERFACE/MOCKUP SYSTEM

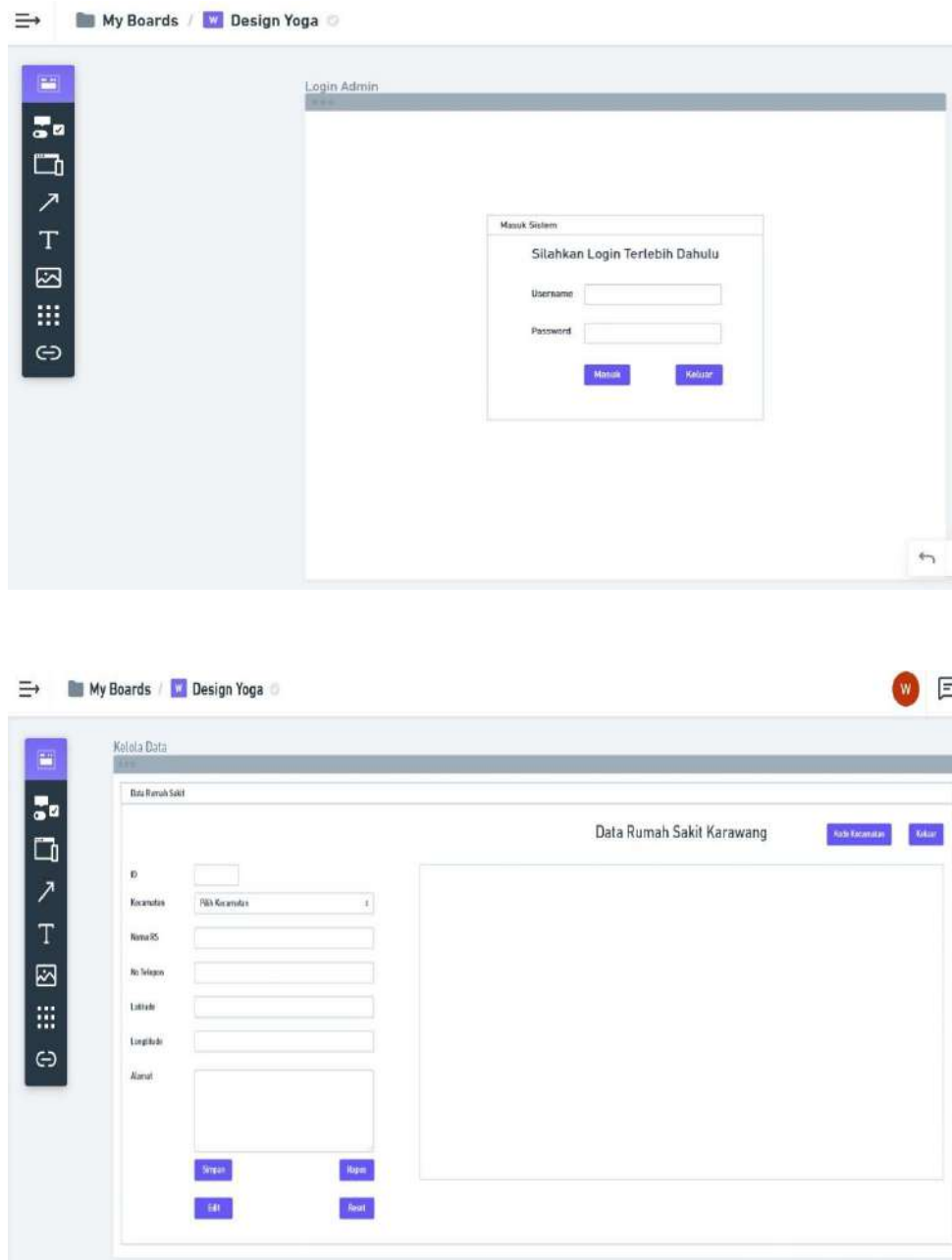
8.1. User Experience

8.1.1. UX User



Gambar 8. 1 UX Untuk User

8.1.2. UX Admin

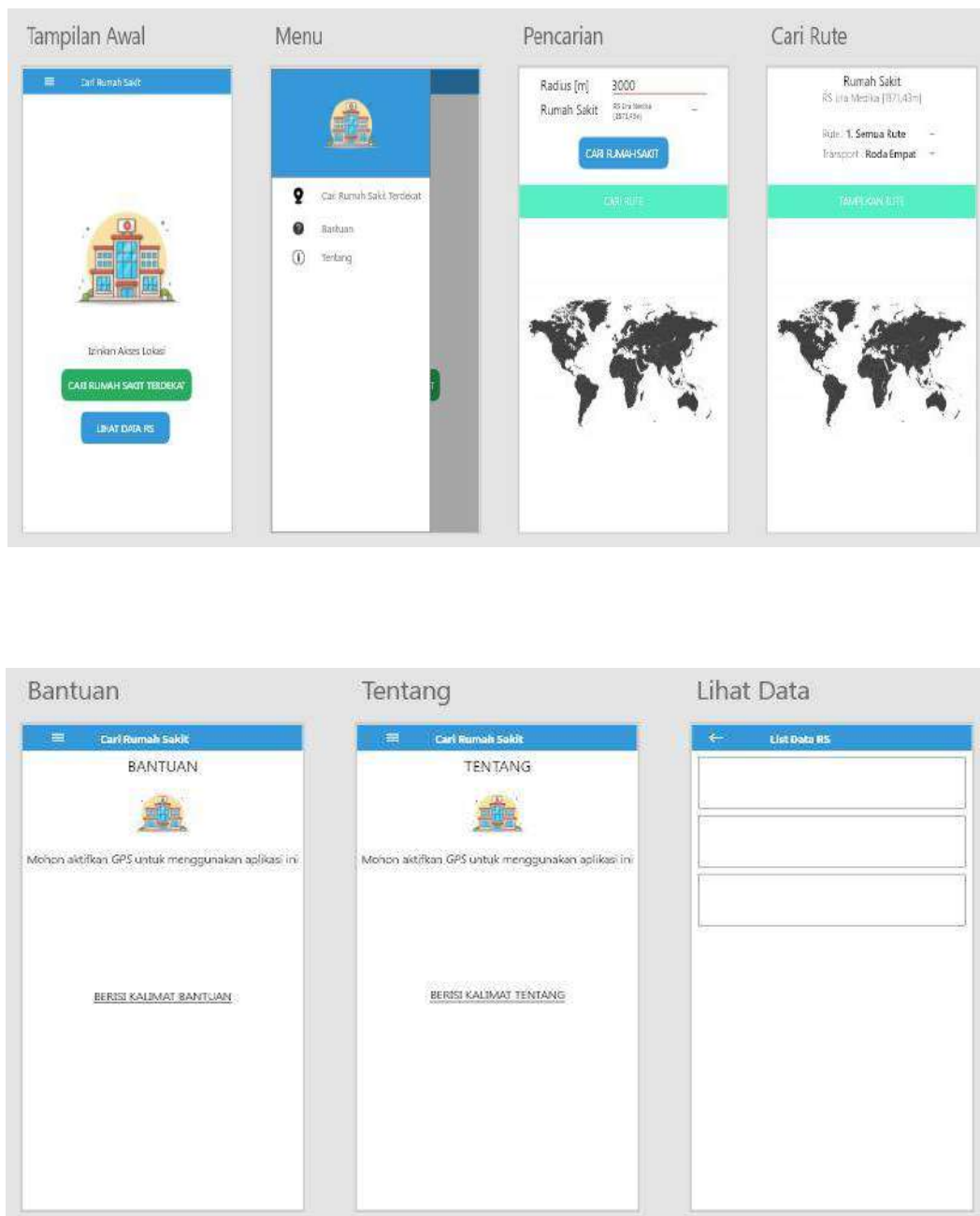


Gambar 8. 2 UX Untuk Admin

Dalam men-desain sebuah *User Experience* menggunakan Whimsical.io.

8.2. User Interface

8.2.1. UI User



Gambar 8. 3 UI Untuk User

8.2.2. UI Admin

Login Admin

Masuk Sistem

Silahkan Login Terlebih Dahulu

Username

Password

Masuk **Keluar**

Kelola Admin

Data Rumah Sakit

Data Rumah Sakit Karawang

Tambah Data **Keluar**

ID:

Kecamatan:

Nama RS:

No Telp:

Latitude:

Longitude:

Alamat:

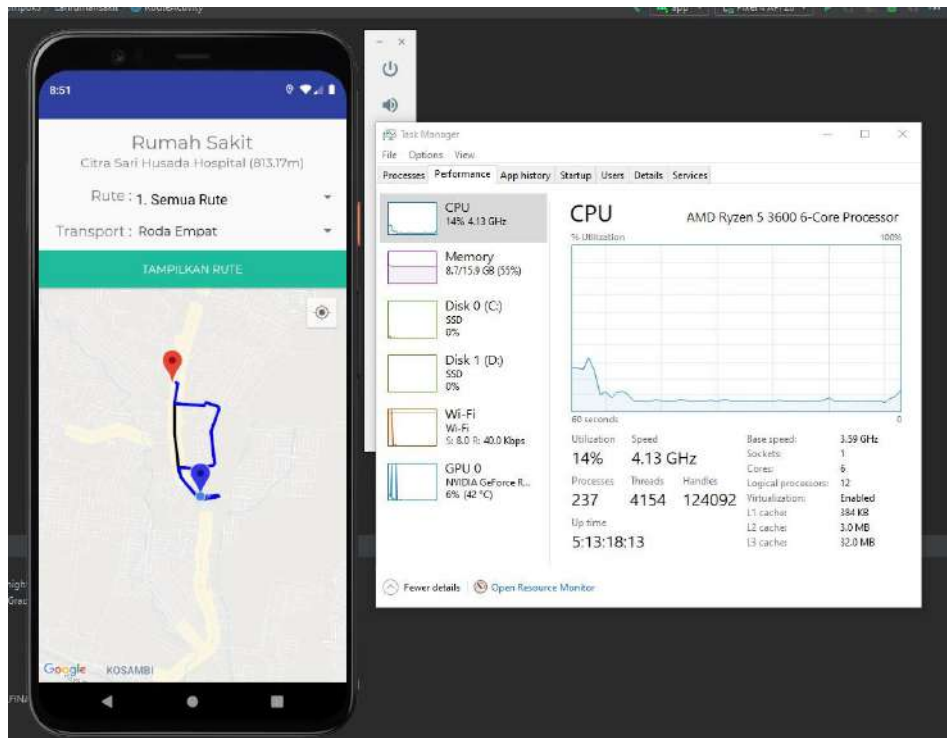
Simpan **Kembali**

ID	Kecamatan	Nama RS	Alamat	No Telp	Latitude	Longitude

Gambar 8. 4 UI Untuk Admin

Dalam men-desain sebuah *User Interface* menggunakan suatu aplikasi yang bernama Adobe Xd dengan beberapa refrensi yaitu Unsplash.com, Flat UI Color, Invision, FreeVector, Remove.bg dan Google Font.

8.3. Performance Aplikasi



Gambar 8. 5 Performance Aplikasi

MODUL 9

UJI PEMAHAMAN PRAKTIKUM

1. Sebuah usecase diagram mempunyai deskripsi dibawah ini :

Sebuah mesin ATM, akan menampilkan “**Masukan PIN**” : jika nasabah meng-**insert** Kartu ATM. Mesin hanya memberikan waktu selama **30 detik** bagi nasabah untuk memasukan **PIN**. Jika waktu terlampaui akan menampilkan “**WAKTU HABIS**” kemudian secara otomatis mesin ATM akan mengeluarkan kartu. Jika PIN valid Nasabah bisa masuk ke dalam system ATM. Jika PIN salah mesin akan mengeluarkan kartu ATM. Mesin ATM akan mengeluarkan kartu ATM jika setelah di insert, Nasabah menekan tombol “**CANCEL**”

Buat Usecase Diagram dan Class Diagram nya.

2. Buatlah sebuah uscase diagram dari kasus sebagai berikut :

Sebuah waduk pembangkit listrik yang sekaligus berfungsi sebagai pengendali banjir dikendalikan oleh sejumlah sensor untuk mendeteksi tinggi muka air. Jika muka air waduk mencapai tinggi maksimum, Sensor akan memerintahkan Pintu Waduk untuk membuka sekaligus mem-Bunyikan Sirine agar penduduk disekitarwaduk waspada akan terjadi peningkatan muka air sungai yang melewati kampung mereka. Jika sensor mendeteksi tinggi muka air telah mencapai kondisi aman sensor akan memerintahkan pintu waduk untuk menutup dan me-Matikan Sirene. Walaupun waduk dikendalikan oleh sensor namun tetap dibutuhkan penjaga aduk yang secara periodik akan mengawasi kondisi waduk, bahkan menjadi tugas Penjaga Waduk untuk meng-Aktifkan sensor atau me-Matikan Sensor.

3. Buatlah Usecase Diagram, Usecase Deskripsi, Class Diagram, Activity Diagram, dan Squence Diagram.

Sistem belanja online melalui internet. Toko online ini yang dikunjungi adalah toko buku dari berbagai penerbit. Pada sistem ini bagi pelanggan yang mendaftar dapat memesan dan menentukan lokasi pengiriman nya.

4. Buatlah Usecase Diagram, Usecase Deskripsi, Class Diagram, Activity Diagram, dan Squence Diagram.

sitem pemeriksaan pasien dipuskesmas yang memiliki beberapa dokter. Setiap pasien dapat berobat kesuatu poliklinik lebih dari satu kali. Setelah diperiksa kesehatannya oleh dokter tertentu, pasien harus membayar pemeriksaan dan obat-obatanyang diterimanya.

DAFTAR PUSTAKA

- Aries Hadi Sutopo. 2002. *Analisis Dan Desain Sistem Berorientasi Objek*. pertama. Yogyakarta: J & J Learning.
- Hamim Tohari. 2015. *ASTAH Analisis Serta Perancangan Sistem Informasi Melalui Pendekatan UML*. Yogyakarta: Andi.
- Munawar. 2018. *Analisis Perancangan Sistem Berorientasi Objek Dengan UML*. Bandung: Teknik Informatika.
- Sholiq, Prof. Dr. Ir. Imam Robandi, M.T (pendamping). 2010. *Analisis Dan Perancangan Berorientasi Objek*. Bandung: Muara Indah.