



UNIVERSITAS IPWIJA

Pemodelan Analisa Dengan Pendekatan Berorientasi Objek





Konsep Pemodelan ?

Pada pengembangannya sistem informasi model biasanya digambarkan dalam bentuk fisik dan abstrak.

Pengembangan software umumnya dilakukan oleh sebuah tim dimana masing-masing orang dalam tim tersebut memerlukan model untuk mendapatkan gambaran dari sistem tersebut.

Walaupun software tersebut dibangun oleh satu orang tetap saja sebuah model diperlukan karena pengembangan software merupakan satu kegiatan yang kompleks.



Model ?

Suatu Gambaran yang merepresentasikan suatu hal, ada berapa pemahaman yang berkaitan dengan model :

- Sebuah Model harus cepat dan mudah dibangun.
- Sebuah model bisa digunakan untuk simulasi, mempelajari mengenai sesuatu yang akan direpresentasikan
- Sebuah model mampu mempelajari perkembangan dari suatu kegiatan atau masalah
- Kita bisa memilih secara rinci sebuah model
- Model bisa merepresentasikan sesuatu secara real atau tidak sebuah domain.



Diagram ?

Diagram merupakan bentuk nyata yang digunakan untuk merepresentasikan sesuatu atau aksi dari dunia nyata.

Fungsi diagram biasanya digunakan oleh analis dan designer untuk :

- Mengkomunikasikan ide-ide
- Generate ide baru serta segala kemungkinan
- Melakukan tes terhadap ide serta membuat prediksi
- Mempelajari struktur dan hubungan suatu sistem



Acuan dalam merancang sebuah model ?

Simplicity representation – Hanya menggambarkan apa yang harus ditampilkan

Internal consistency – Pada sekumpulan diagram

Completeness – Menampilkan semua yang dibutuhkan

Hierarchical representation – Dapat diturunkan untuk melihat lebih detail pada level yang lebih rendah.



Unified Modeling Language ?

Menurut Boocm dkk.

UML adalah Bahasa standar untuk menulis blue print PL. UML dapat digunakan untuk memvisualisasikan, menentukan, membuat dan mendokumentasikan artefak dari sistem PL secara intensif.

UML sesuai untuk sistem pemodelan mulai dari sistem informasi perusahaan, aplikasi berbasis web yang terdistribusi, bahkan sampai sistem real time embedded yang sulit.

UML diharapkan mampu mempermudah pengembangan piranti lunak (RPL) serta memenuhi semua kebutuhan pengguna dengan efektif, lengkap dan tepat.

Manfaat UML

- Dapat memberikan bahasa pemodelan visual atau gambar kepada para pengguna dari berbagai macam pemrograman maupun proses umum rekayasa.
- Menyatukan informasi-informasi terbaik yang ada dalam pemodelan.
- Memberikan suatu gambaran model atau sebagai bahasa pemodelan visual yang ekspresif dalam pengembangan sistem.
- Tidak hanya menggambarkan model sistem software saja, namun dapat memodelkan sistem berorientasi objek.
- Mempermudah pengguna untuk membaca suatu sistem.
- Berguna sebagai blueprint, jelas ini nantinya menjelaskan informasi yang lebih detail dalam perancangan berupa coding suatu program.

UML **vs** Flowchart ?

UML

- Untuk penggambaran aliran aktivitas dalam sistem dengan simbol.
- Bagaimana masing-masing berawal, bagaimana mereka berinteraksi, apa saja yang mereka bisa lakukan, dll.
- UML digunakan pada sistem dengan paradigma berorientasi objek.

Flowchart

- Untuk penggambaran aliran algoritma didalam program atau prosedur program secara logika dengan simbol.
- Flowchart digunakan untuk mendeskripsikan langkah-langkah penyelesaian masalah pada program atau prosedur program.
- Flowchart digunakan pada program dengan paradigma prosedural.

UML **vs** Flowchart ?

Contoh kasus :

Program Belajar Daring Kampus

UML

- User yang terdapat didalam sistem : Mahasiswa, Dosen, Staff, dll.
- Data yang diperlukan : Mata Kuliah, Gedung, dll.
- Mahasiswa mempunyai data : Nama, NPM, Jurusan, dll.
- Yang dilakukan dosen : Mengajar, Memberikan nilai, dll.
- dll...

Flowchart

- Alur kerja login user : input data login, jika data login == salah satu data user maka berhasil, dll.
- Alur kerja upload materi Dosen : input dokumen, validasi dokumen, jika dokumen diperbolehkan maka tampil “materi berhasil diupload”.
- dll...

UML adalah bahasa untuk:

a. Visualizing

Beberapa hal dimodelkan secara tekstual atau dengan model grafis.

UML adalah Bahasa grafis yang menggunakan sekelompok simbol grafis.

Setiap simbol dalam notasi UML didefinisikan dengan baik secara semantic, sehingga pengembang dapat menulis model UML dan dapat menafsirkan model itu dengan jelas.

b. Specifiying

UML dapat membangun model yang tepat, tidak ambigu dan lengkap.

UML membahas spesifikasi semua keputusan analisa, perancangan dan Implementasi penting yang harus dilakukan dalam mengembangkan dan menerapkan sistem PL yang intensif.



UML adalah bahasa untuk:

c. Constructing

UML bukan Bahasa pemrograman visual, namun modelnya bisa langsung terhubung ke berbagai bahasa pemrograman.

UML memungkinkan untuk memetakan ke bahasa pemrograman seperti java, C++, atau bahkan ke table dalam basis data relasional atau penyimpanan database berorientasi objek yang tetap.

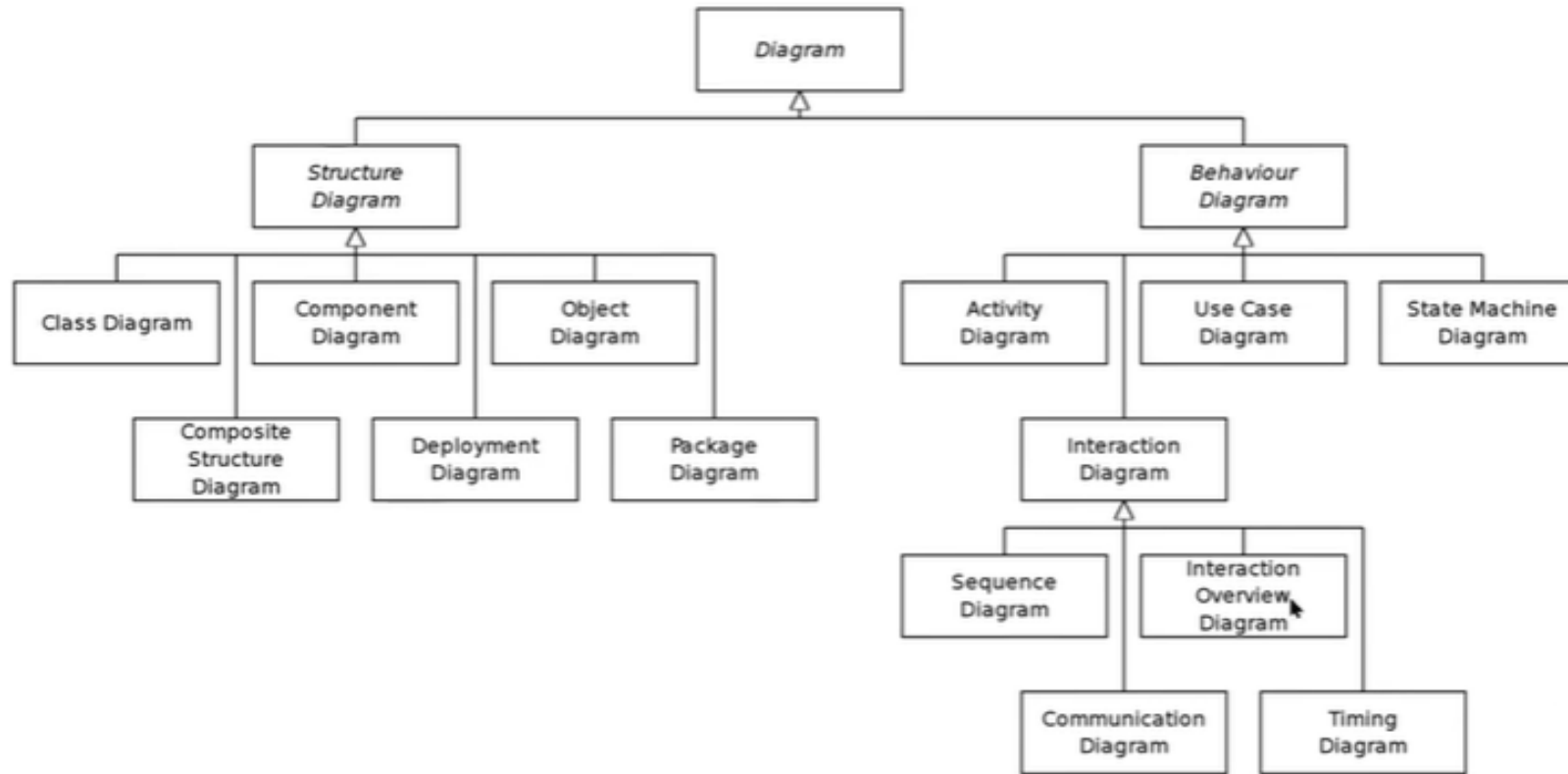
d. Documenting

UML membahas dokumentasi arsitektur sistem dan semua detailnya.

UML menyediakan bahasa untuk mengekspresikan persyaratan dan tes.

UML juga menyediakan bahasa untuk memodelkan kegiatan perencanaan proyek.

Macam-macam Diagram UML ?



Macam-macam Diagram UML ?

- **Structure Diagram** : untuk menjelaskan suatu struktur statis dari sistem yang dimodelkan.
 - **Class Diagram** : untuk memodelkan struktur kelas (class).
 - **Component Diagram** : untuk memodelkan komponen object.
 - **Object Diagram** : untuk memodelkan struktur object.
 - **Composite Structure Diagram** : untuk memodelkan struktur internal dari (component, class, dan use case), termasuk hubungan pengklasifikasian ke bagian lain dari sebuah program.
 - **Deployment Diagram** : untuk memodelkan distribusi aplikasi.
 - **Package Diagram** : untuk memodelkan pengumpulan kelas dan juga menunjukkan bagaimana elemen model akan disusun serta menggambarkan ketergantungan antara paket-paket.

Macam-macam Diagram UML ?

- **Behavior Diagram** : untuk menjelaskan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
 - Activity Diagram : untuk memodelkan perilaku Use Cases dan objects di dalam sistem.
 - Use Case Diagram : untuk memodelkan proses bisnis.
 - State Diagram : untuk memodelkan perilaku objects di dalam sistem.

Macam-macam Diagram UML ?

- **Interaction Diagram** : untuk menjelaskan interaksi sistem dengan sistem lain maupun antar sistem pada sebuah sistem.
 - Sequence Diagram : untuk memodelkan pengiriman pesan (message) antar objects.
 - Interaction Overview Diagram : untuk memodelkan hubungan dan kerjasama antara activity diagram dengan sequence diagram.
 - Communication Diagram : untuk menjelaskan proses terjadinya suatu aktivitas dan diagram ini juga menggambarkan interaksi antara objek yang ada pada sebuah sistem.
 - Timming Diagram : untuk memodelkan bentuk lain dari interaksi diagram, dimana fokus yang paling utamanya kepada waktu.



1 Use Case Diagram

- Use Case Diagram digunakan untuk menggambarkan serangkaian tindakan *Use Case* bahwa sistem dapat melakukan interaksi di luar sistem (aktor) dengan sistem itu sendiri (abstraksi).
- Use Case juga digunakan untuk mengetahui fungsi apa saja yang ada dalam sebuah sistem dan siapa yang berhak menggunakan fungsi-fungsi itu.
- Nama Use Case didefinisikan semudah mungkin dan dapat dipahami.
- Dua hal utama pada case yaitu Pendefinisikan Actor dan Use Case.

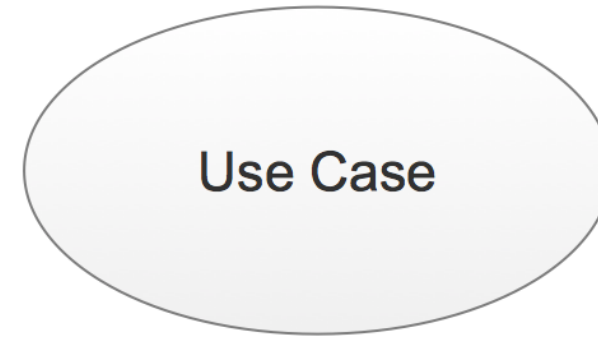
A person in a dark suit is shown from the chest down, stacking light-colored wooden blocks on a wooden table. The person's hands are visible, with one hand holding a block and the other supporting the stack. The background is a soft, out-of-focus grey.

Use Case Diagram

- Use Case Diagram digunakan untuk:
 - a. Merepresentasikan interaksi sistem **Pengguna**
 - b. Medefinisikan dan mengatur persyaratan fungsional dalam suatu sistem
 - c. Menentukan konteks dan persyaratan sistem.
 - d. Memodelkan aliran dasar peristiwa dalam Use Case

Simbol Use case

Use Case



- Digambarkan dengan eclips horizontal
- Nama Use Case menggunakan kata kerja

Simbol Use case

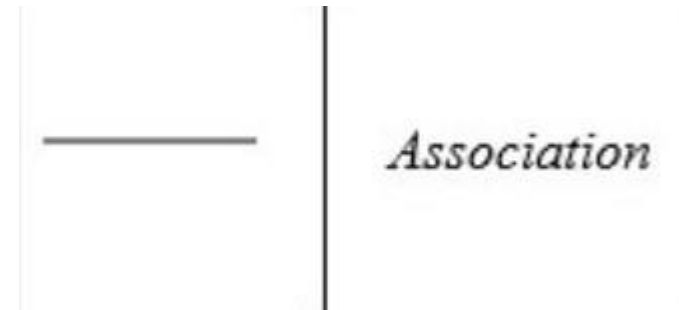
Actor



- Menggambarkan orang, system/external entitas yang menyediakan atau menerima informasi
- Merupakan lingkungan luar dari sistem
- Aktor utama digambarkan pada pojok kiri atas dari sebuah diagram

Simbol Use case

Asosiasi



- Menggambarkan **bagaimana Actor** berinteraksi dengan Use Case
- **Bukan** menggambarkan aliran data/informasi

Simbol Use case

Generalisasi

—————→ Generalization

- Menggambarkan **generalisasi** antar Use Case atau antara Actor dengan panah penutup yang mengarah dari **child** ke **parent**.

Simbol Use case

Relasi Include

`<include>` → Include

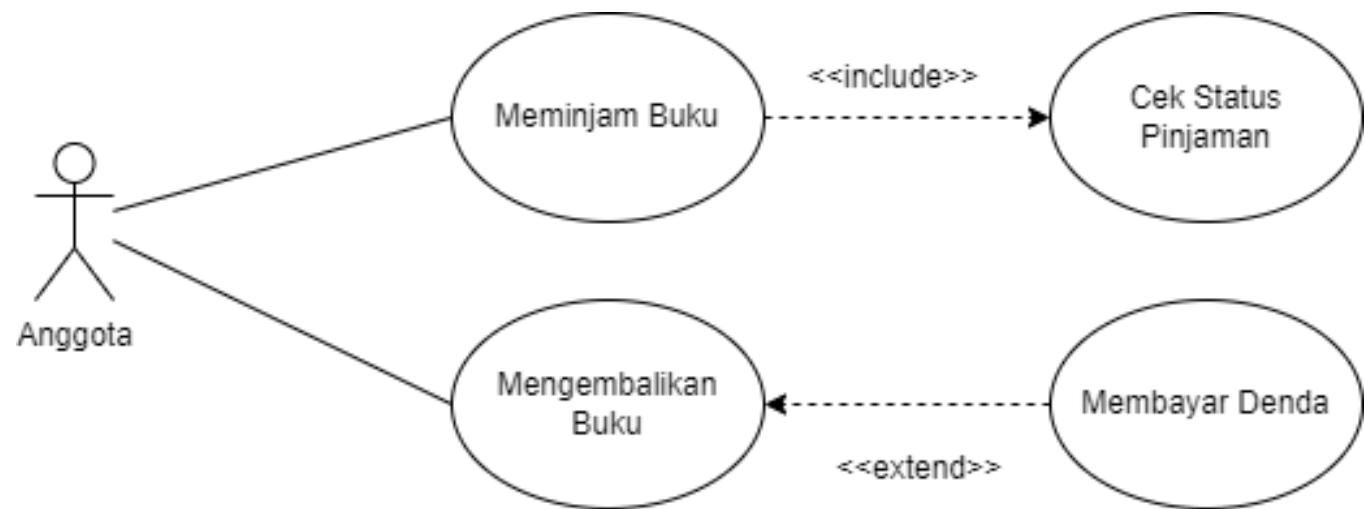
- Hubungan antara dua Use Case untuk menunjukkan adanya perilaku Use Case yang dimasukkan ke dalam perilaku dari **Base Use Case**
- Tanda panah terbuka harus terarah ke **Sub Use Case**

Relasi Extend

`<extend>` → Extend

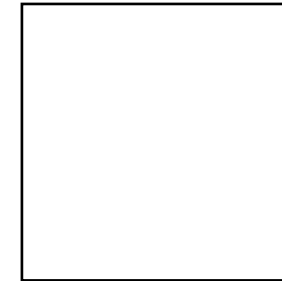
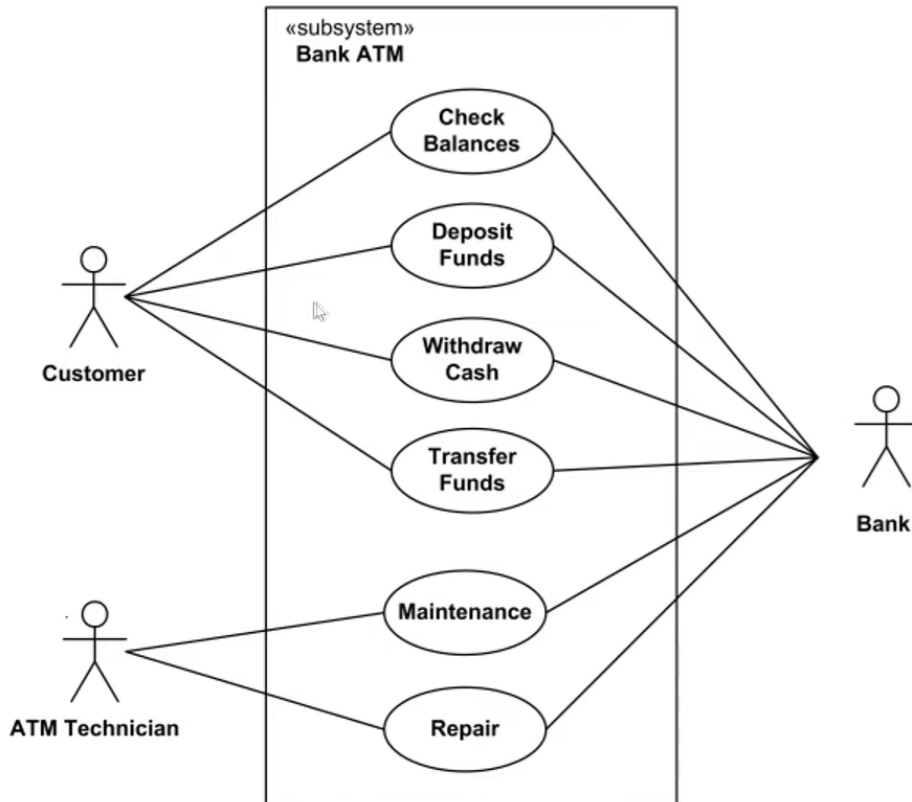
- Perluasan dari Use Case lain (*optional*)
- Tanda panah terbuka harus terarah ke **Base Use Case**

Contoh penggunaan relasi include dan extend



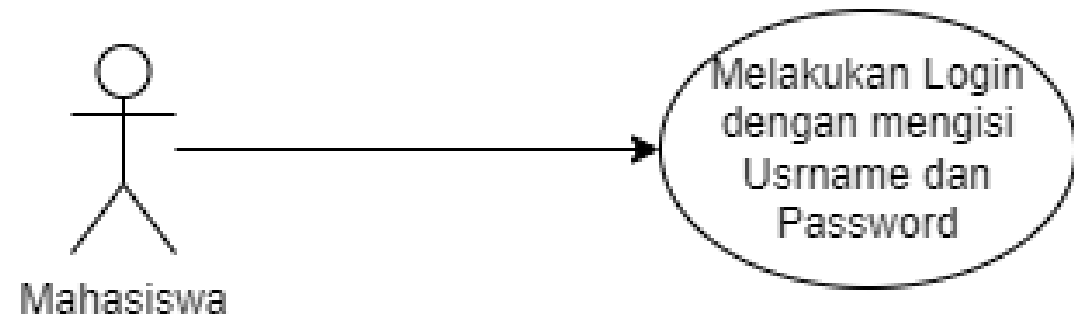
Simbol Use case

Boundary Boxes



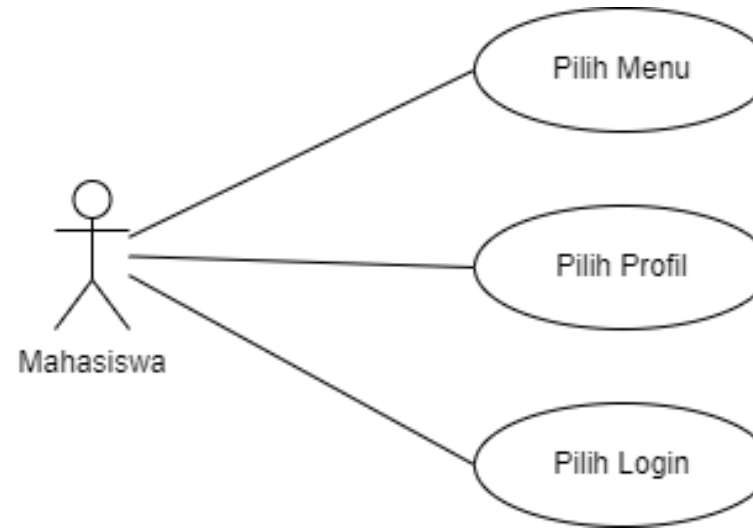
- Untuk memperlihatkan **Batasan Sistem** dengan lingkungan luar sistem

Contoh Penggambaran Use Case



- Pada gambar di atas, penulisan nama use case **tidak dituliskan secara berlebihan**, seharusnya hanya bisnis proses saja yang dituliskan yaitu LOGIN
- Asosiasi actor dengan use case sebaiknya tidak menggunakan tanda panah, kecuali untuk penggunaan relasi include/extend, dan generalisasi

Contoh Penggambaran Use Case



Pada gambar di atas, penulisan nama use case **Pilih Menu**, **Pilih Profil**, **Pilih Login**, tidak seharusnya ditulis demikian.

Use case dituliskan dengan bisnis proses, bukan asal seperti dalam Activity Diagram



Use Case Pembelian Tiket

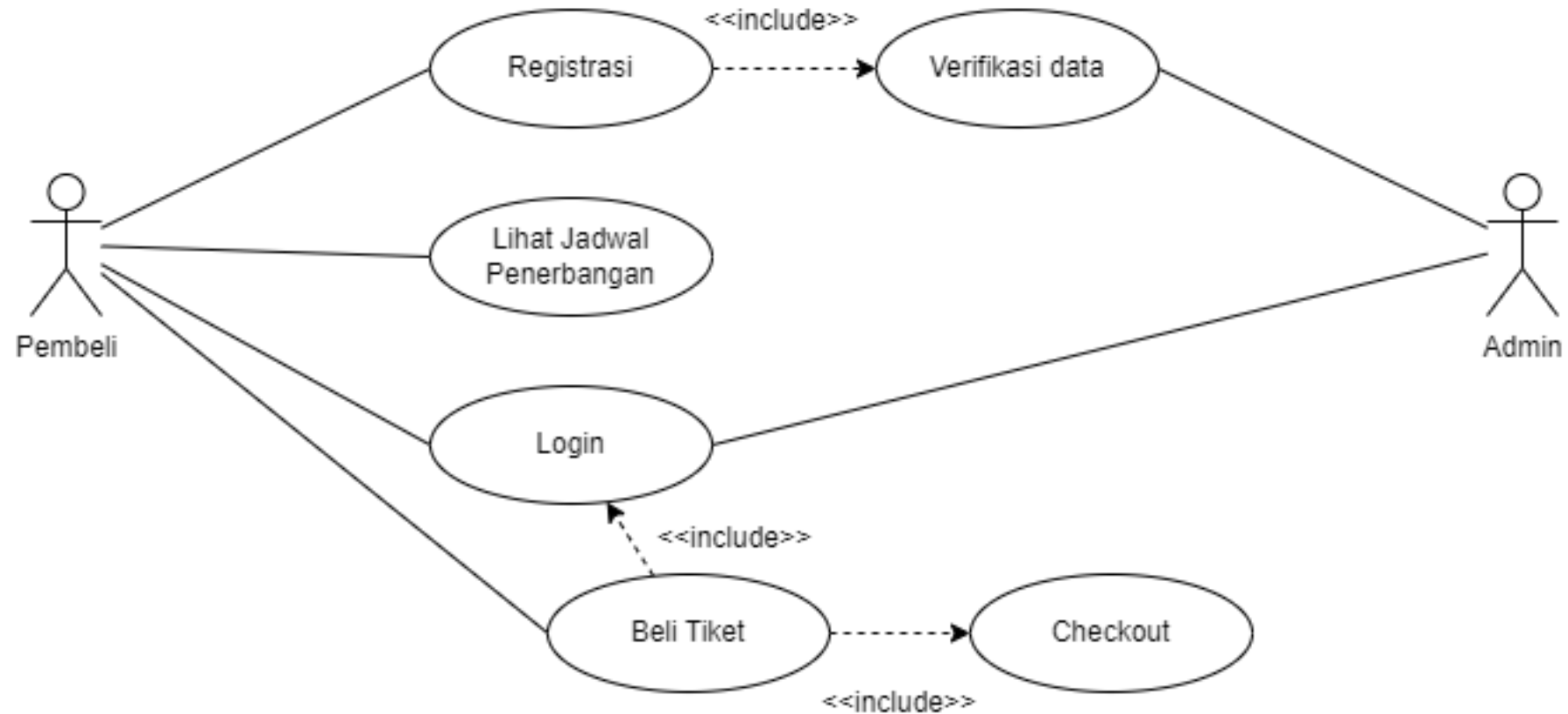
1.A Daftar Kebutuhan Fungsional Use Case

NO	Use Case	Aktor	Deskripsi
1	Registrasi	Pembeli	Use Case ini berfungsi untuk proses pendaftaran sebagai pembeli
2	Verifikasi Data	Admin	Use Case ini berfungsi untuk melakukan pengecekan data pembeli
3	Lihat Jadwal Penerbangan	Pembeli	Use Case ini berfungsi untuk melihat jadwal penerbangan oleh pembeli
4	Login	Pembeli, Admin	Use case ini berfungsi untuk masuk kedalam system untuk pembelian dan admin
5	Beli tiket	Pembeli	Use case ini berfungsi untuk membeli tiket penerbangan
6	Checkout	Pembeli	Use case ini berfungsi untuk pembeli melakukan pembayaran pembelian tiket

1.B Daftar Kebutuhan Fungsional Aktor

NO	Aktor	Deskripsi
1	Pembeli	Pembeli adalah actor yang dapat melakukan registrasi, melihat jadwal penerbangan, melakukan login, membeli tiket dan checkout
2	Admin	Admin adalah actor yang dapat melakukan verifikasi data pembelian melakukan login

2. Use Case Pembelian Tiket Online



A person in a dark suit is shown from the chest down, carefully placing a wooden block onto a stack of other blocks on a wooden table. The blocks are light-colored and rectangular. The background is a soft, out-of-focus grey.

3. Spesifikasi Use Case

Nama Use Case	Beli Tiket
Deskripsi	Use Case ini menyediakan layanan pembelian tiket penerbangan
Aktor	Pembeli
Pre-Condition	Login
Post-Condition	<ul style="list-style-type: none">• Pembeli melengkapi data penumpang• Pembeli memilih cara pembayaran• Pembeli melakukan checkout
Relasi	Include dari login

A person in a dark suit is shown from the chest down, stacking light-colored wooden blocks on a wooden table. The person's hands are visible, and they are in the process of placing a block on top of a stack. The background is a soft, out-of-focus grey.

3. Skenario Use Case

Aksi Aktor	Reaksi Sistem
1. Pembeli melakukan registrasi	
	2. Sistem menyimpan data registrasi
3. Pembeli melihat jadwal penerbangan	
	4. Sistem akan menampilkan form data penumpang yang harus diisi oleh pembeli
5. Pembeli membeli tiket penerbangan	
	6. Sistem menampilkan form data penumpang yang harus diisi oleh pembeli
7. Pembeli mengisi data penumpang	
	8. Sistem menampilkan halaman untuk memilih cara pembayaran



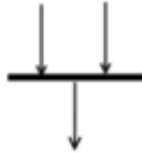
3. Skenario Use Case

Aksi Aktor	Reaksi Sistem
9. Pembeli memilih cara pembayaran	
	10. Sistem menampilkan halaman checkout 11. Sistem mengirimkan notifikasi pembelian tiket
12. Pembeli melakukan pembayaran 13. Pembeli melakukan konfirmasi pembayaran	
	14. Sistem mengirimkan e-tiket melalui email

2 Activity Diagram

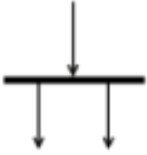


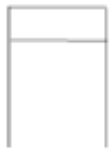
- Activity diagram adalah Teknik untuk menggambarkan logika procedural, proses bisnis dan jaringan kerja antara pengguna dan system.
- Menggunakan notasi yang **mirip flowchart**, meskipun terdapat sedikit perbedaan notasi karena diagram ini mendukung behavior paralel.
- Activity diagram dibuat berdasarkan **sebuah atau beberapa use case** pada use case diagram.
- Memungkinkan melakukan proses untuk memilih urutan dalam melakukannya atau hanya menyebutkan aturan-aturan rangkaian dasar yang haru diikuti, karena proses-proses sering muncul secara paralel.

Simbol Activity Diagram

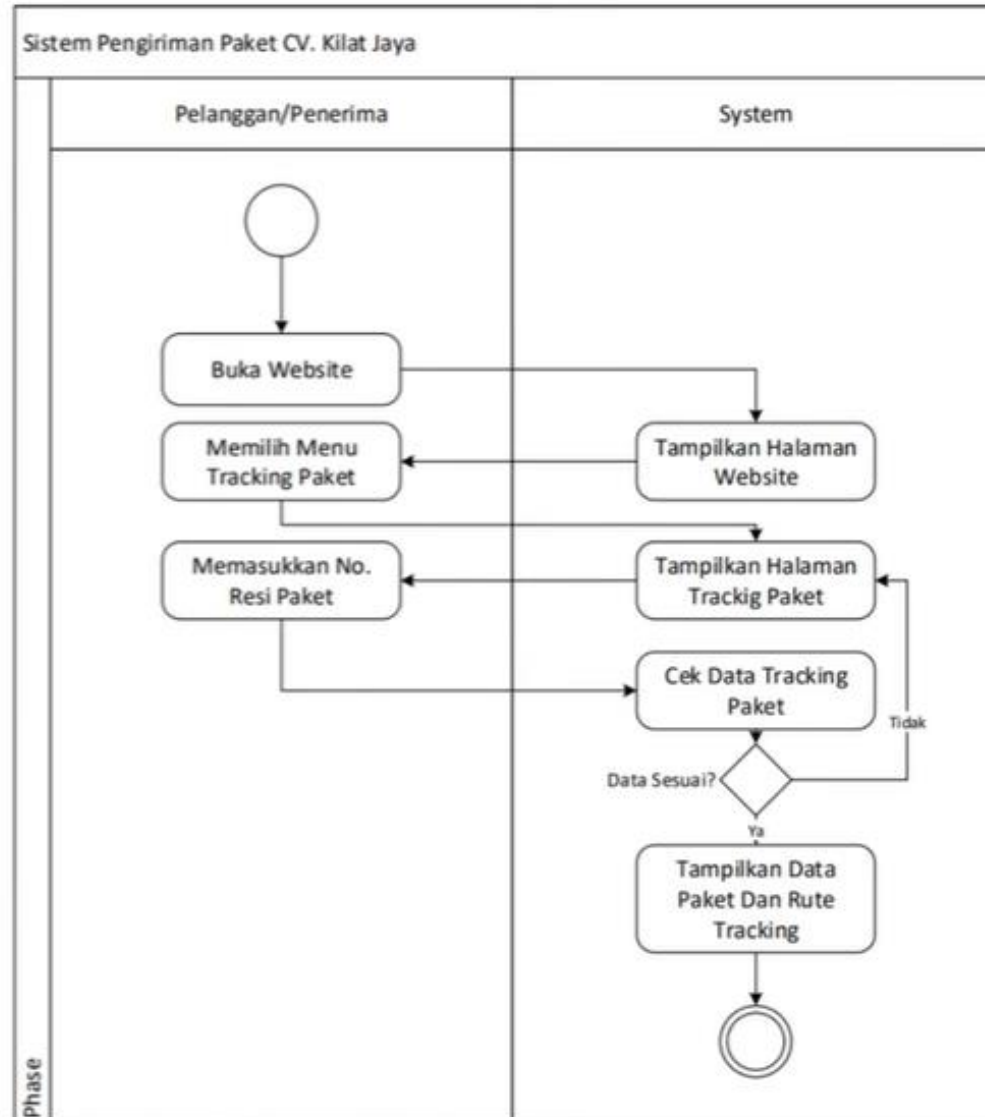
No	Nama	Simbol	Keterangan
1.	Start	●	<ul style="list-style-type: none">• Menjelaskan awal proses kerja dalam activity diagram• Hanya ada satu simbol start
2.	End	⦿	<ul style="list-style-type: none">• Menandai kondisi akhir dari suatu aktivitas dan merepresentasikan penyelesaian semua arus proses• Bisa lebih dari satu simbol end
3.	Activity	□	<ul style="list-style-type: none">• Menunjukkan kegiatan yang membentuk proses dalam diagram
4.	Join		<ul style="list-style-type: none">• Menggabungkan dua atau lebih aktivitas bersamaan dan menghasilkan hanya satu aktivitas yang terjadi dalam satu waktu



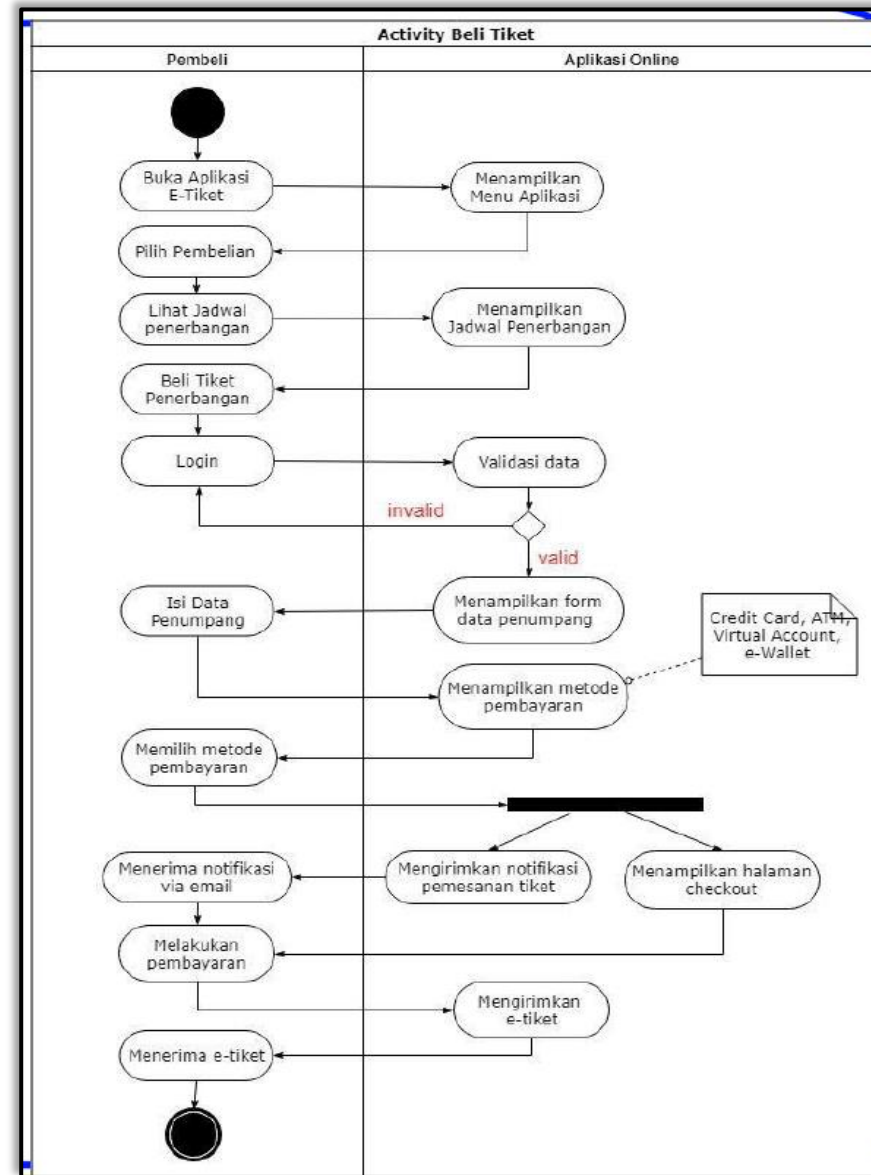
Simbol Activity Diagram

5.	Fork		<ul style="list-style-type: none">• Membagi aliran aktivitas tunggal menjadi beberapa aktivitas bersamaan
6.	Decision		<ul style="list-style-type: none">• Mewakili keputusan yang memiliki setidaknya dua jalur bercabang yang kondisinya sesuai dengan opsi pencabangan
7.	Connector		<ul style="list-style-type: none">• Menunjukkan arah aliran atau aliran kontrol dari aktivitas
8.	Swimlane		<ul style="list-style-type: none">• Cara untuk mengelompokkan aktivitas berdasarkan aktor• Menggunakan garis vertikal

Contoh Activity Diagram (1)



Contoh Activity Diagram (2)





Software UML

- draw.io: <https://app.diagrams.net/>
- Ms. Visio
- dll

A person in a dark suit and tie is shown from the chest down, stacking light-colored wooden blocks on a wooden table. The person's hands are visible, with one hand holding a block and the other supporting the stack. The background is a soft, out-of-focus grey.

Tugas Pengantar RPL (2) :

Buatlah sebuah use case diagram dengan studi kasus penerimaan mahasiswa online dengan ketentuan sebagai berikut :

- Buat Daftar Kebutuhan Use case
- Buat Daftar Kebutuhan Fungsional Aktor
- Buat Use Case Diagram
- Buat Spesifikasi Use case
- Buat Skenario Use case

*Buat dalam bentuk PDF
Submit max. 31 Oktober 2023*

*File diagram (draw.io / viso) dikirimkan email: kuliahpersis@gmail.com
Subject email: Tugas RPL UML 2_Nama*