OOAD (Object Oriented Analysis and Design)
UML
part 1 (Usecase)

Gentisya Tri Mardiani, S.Kom., M.Kom

ADSI-2017

OOAD (Object Oriented Analysis and Design)

- Salah satu pendekatan analisis dan desain yang dapat digunakan selain terstruktur

- Pendekatan yang dapat digunakan oleh pengembang software yang menitik beratkan solusi berdasarkan objek beserta relasinya

Objek

- Objek adalah benda di dunia nyata yang dapat dibedakan satu dengan yang lainnya
- Objek dapat dibentuk dari domain permasalahan yang diambil
- Objek mempunyai identitas, properti, dan tingkah laku
- Objek merupakan hasil inisiasi dari kelas

Kelas

- Sekumpulan objek yang memiliki kemiripan dalam hal atribut, properti, behaviour, dan semantik

- Proses klasifikasi dilakukan untuk membentuk kelompok dari beberapa objek yang memiliki kemiripan

Atribut

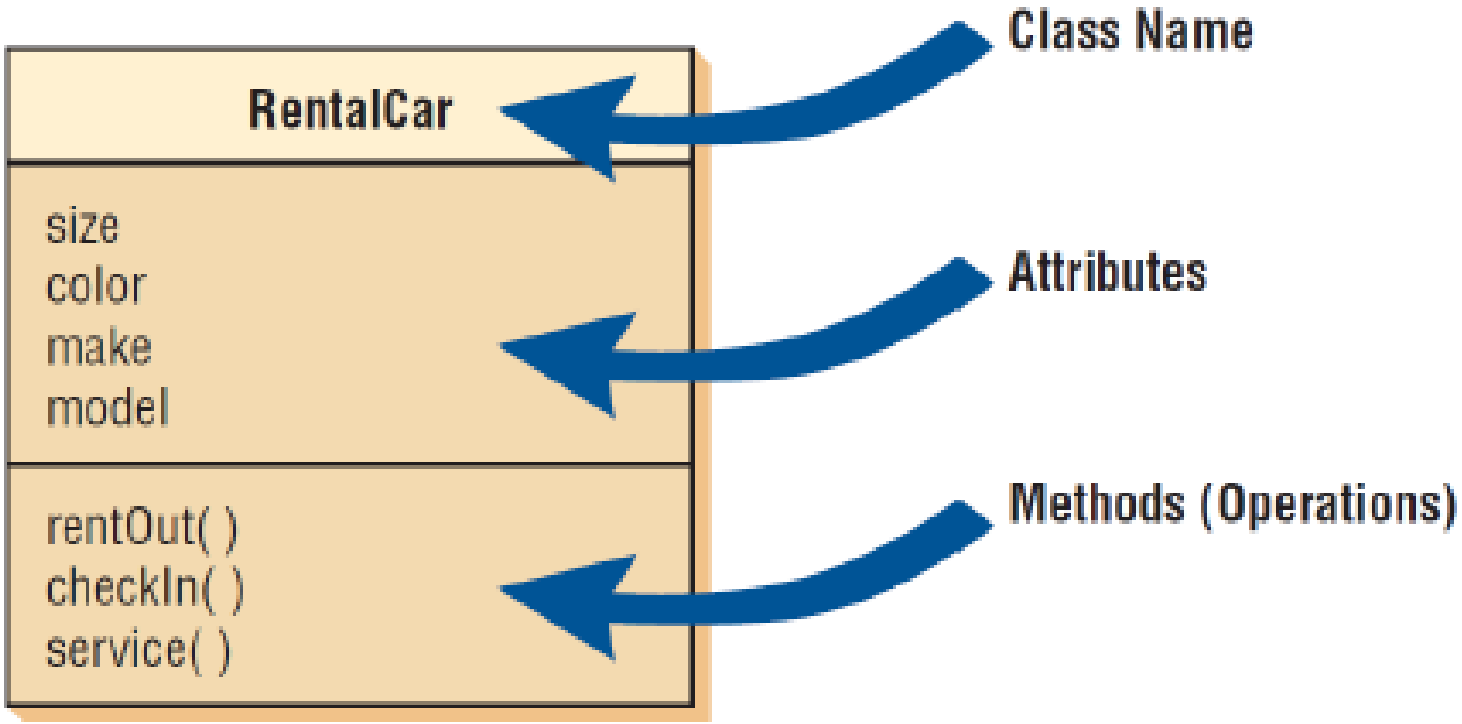
- Atribut adalah data yang membedakan antara objek satu dengan yang lainnya.
- Dalam class, atribut sering disebut sebagai variabel. Atribut dibedakan menjadi dua jenis yaitu ***Instance Variable*** dan ***Class Variable***.

- ***Instance variable*** adalah atribut untuk tiap objek dari kelas yang sama. Tiap objek mempunyai dan menyimpan nilai atributnya sendiri. Jadi, tiap objek dari class yang sama boleh mempunyai nilai yang sama atau berbeda.
- ***Class Variable*** adalah atribut untuk semua objek yang dibuat dari class yang sama. Semua objek mempunyai nilai atribut yang sama. Jadi semua objek dari class yang sama mempunyai hanya satu nilai yang value nya sama.

Behaviour

- Behaviour/tingkah laku adalah hal-hal yang bisa dilakukan oleh objek dari suatu class.
- Behaviour dapat digunakan untuk mengubah nilai atribut suatu objek, menerima informasi dari objek lain, dan mengirim informasi ke objek lain untuk melakukan suatu tugas (task).
- Dalam class, behavior disebut juga sebagai **methods**.
- Methods adalah serangkaian statements dalam suatu class yang menghandle suatu task tertentu.
- Cara objek berkomunikasi dengan objek yang lain adalah dengan menggunakan method.

Kelas


RentalCar
size
color
make
model
rentOut( )
checkIn( )
service( )
Class Name
Attributes
Methods (Operations)

Tipe diagram UML

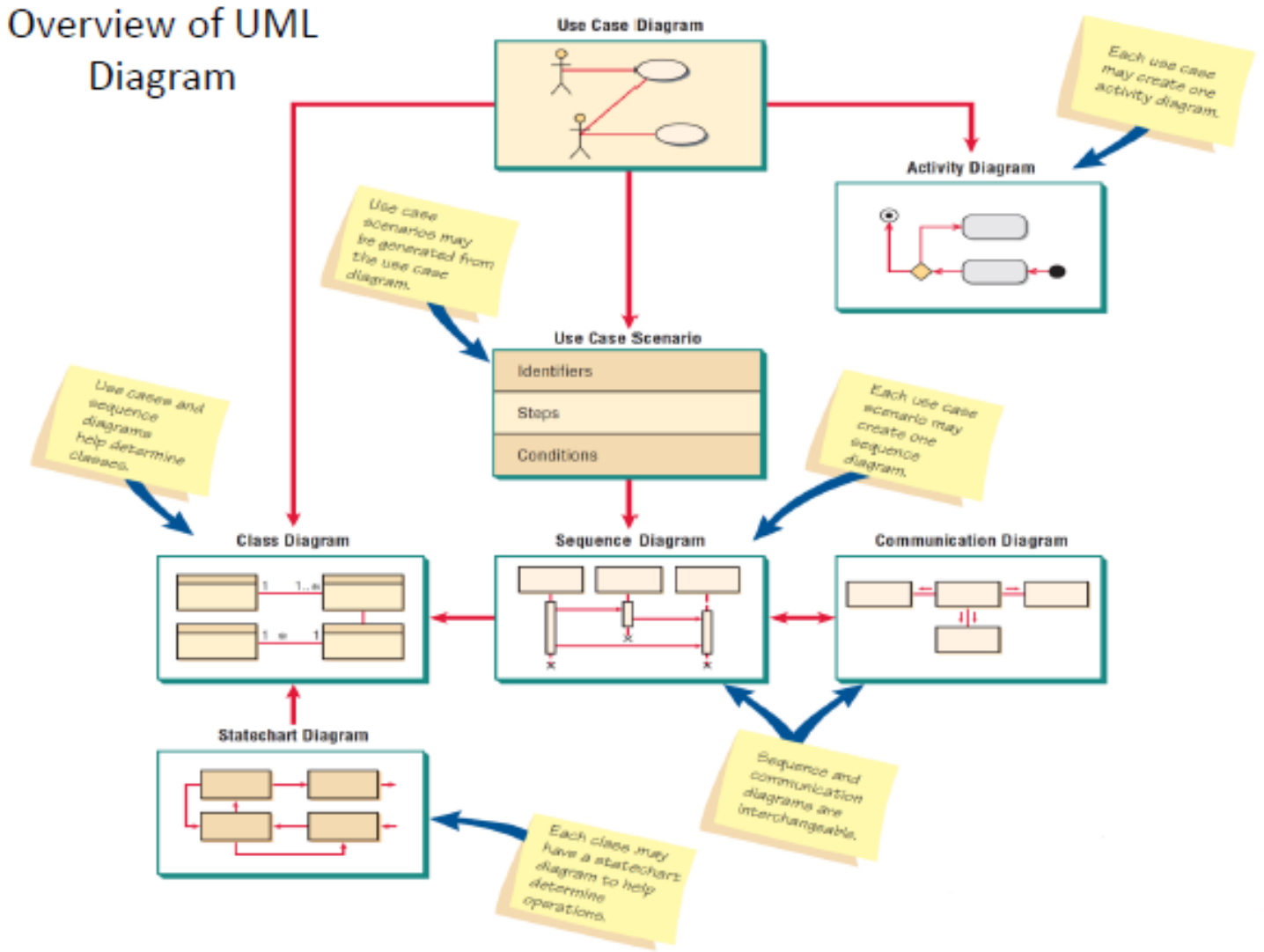
- **Structural diagrams –** digunakan untuk mendeskripsikan relasi antar kelas (class diagram, component diagram, deployment diagram)

- **Behaviour diagrams –** digunakan untuk mendeskripsikan interaksi antar aktor dan usecase (usecase diagram, activity diagram, sequence diagram, collaboration diagram, statechart diagram)

diagram UML yang sering digunakan

1. **Usecase diagram**
2. **Usecase scenario**
3. **Activity diagram**
4. **Sequence diagram**
5. **Class diagram**
6. **Statechart diagram**

Relasi diagram UML



USE CASE DIAGRAM

Konsep Use case diagram

- **Tujuan:** menggambarkan dan mendokumentasikan persyaratan sistem dari sudut pandang pengguna dengan cara yang mudah dipahami.
- Proses pemodelan fungsi-fungsi sistem dalam konteks peristiwa-peristiwa bisnis, siapa yang mengawalinya, dan bagaimana sistem itu merespon.
- Menggambarkan **apa** yang dilakukan sistem, bukan **bagaimana** sistem melakukannya

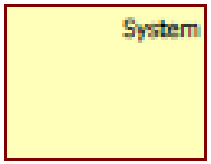
Use case diagram

- Diagram yang menggambarkan interaksi antara sistem dengan sistem eksternal dan pengguna.

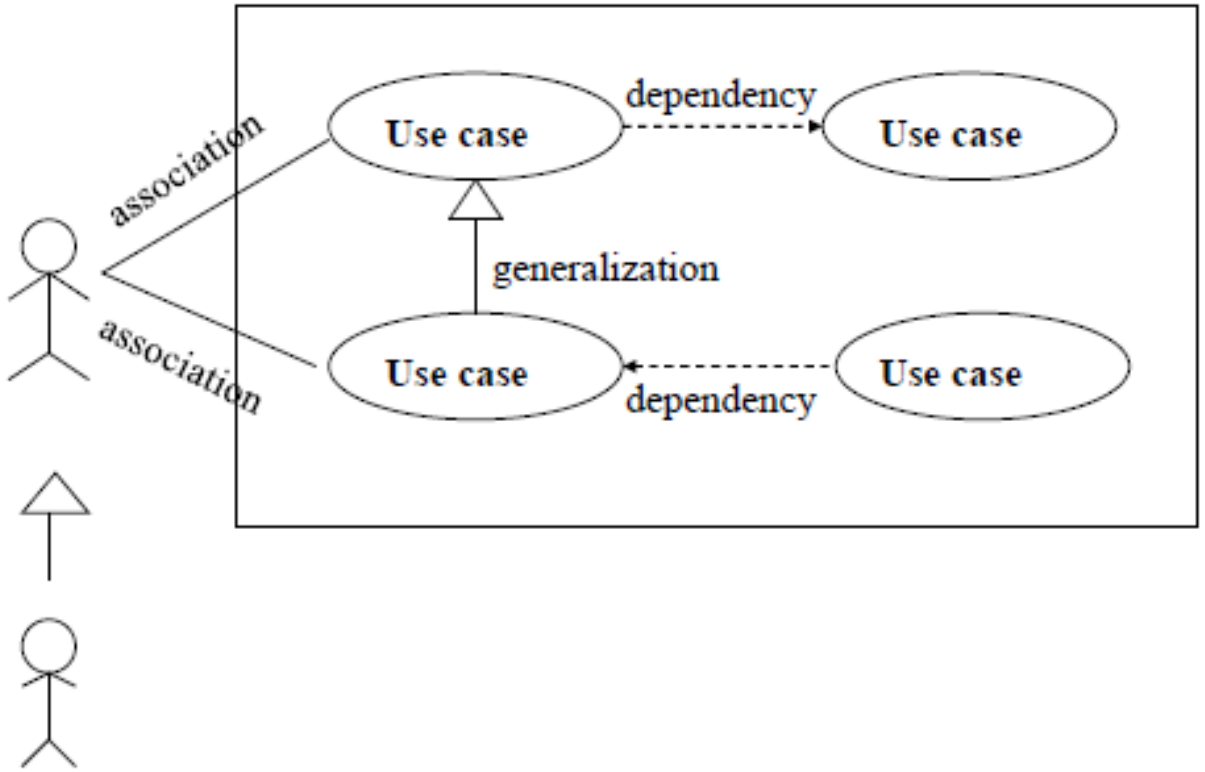
- Secara grafis menggambarkan siapa yang akan menggunakan sistem dan dengan cara apa pengguna mengharapkan untuk berinteraksi dengan sistem.

- Mendeskripsikan apa yang sistem lakukan tanpa mendeskripsikan bagaimana sistem melakukaknnya

Simbol Use case diagram

SIMBOL	NAMA SIMBOL	FUNGSI
	Aktor	Pihak yang mengakses use case
	Use Case	Mewakili apa yang sistem bisa lakukan
	Association	Merelasikan aktor dengan use case
	System Boundary	Menggambarkan batasan sistem terhadap lingkungannya

Use case diagram



Aktor

- Aktor adalah segala hal diluar sistem yang akan menggunakan sistem tersebut untuk melakukan sesuatu (Kurt Bittner, Ian Spence. 2002).
- Cara mudah untuk menemukan aktor adalah dengan bertanya hal-hal berikut:
  - **SIAPA** yang akan menggunakan sistem?
  - **APAKAH** sistem tersebut akan memberikan **NILAI bagi aktor**?

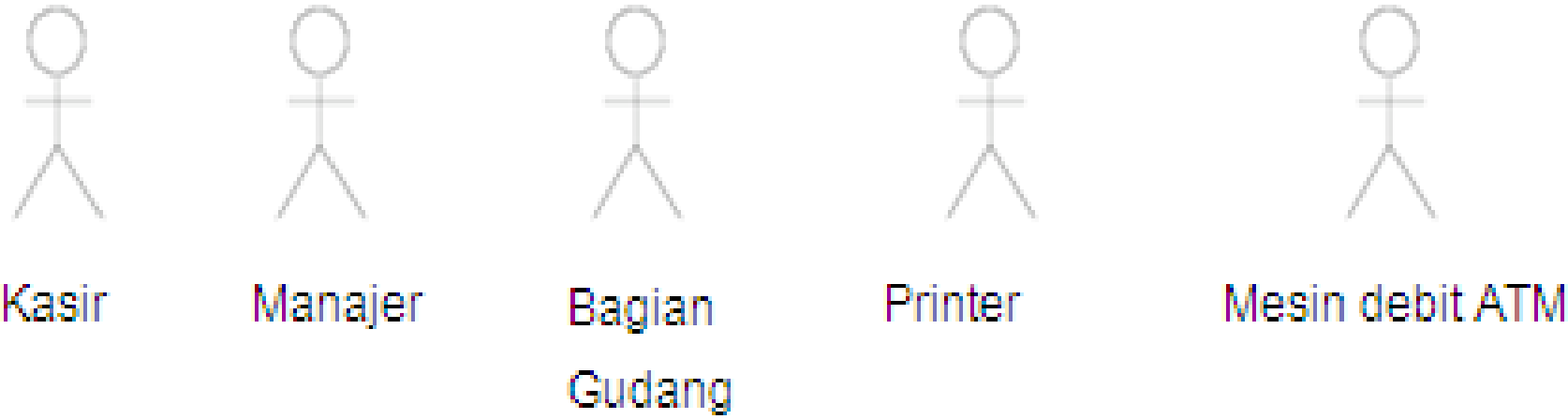
Aktor

- Tidak semua aktor adalah manusia, bisa saja sistem lain yang berinteraksi dengan sistem yang dibangun.
- Pertimbangan untuk menemukan sistem lain sebagai aktor, yaitu:
  - Jika sistem bergantung pada sistem lain untuk melakukan sesuatu, maka sistem lain itu adalah aktor.
  - Jika sistem lain itu meminta (*request*) informasi dari sistem yang akan dibangun, maka sistem lain itu adalah aktor
- Penamaan aktor sesuai dengan PERANnya

Pertanyaan	Analisis
Siapa sajakah yang berinteraksi dengan sistem pencatatan penjualan di supermarket?	• Bagian yang akan mencatat penjualan barang • Bagian yang ingin tahu berapa besar keuntungan yang didapatkan • Bagian yang ingin tahu berapa banyak produk yang berkurang
Peran apa saja yang terlibat?	Kasir, manajer, bagian gudang.
Nilai apa sajakah yang akan diberikan sistem kepada aktor?	Nilai bagi kasir: • Ia akan mendapatkan struk belanja. • Lama aktivitas kerja akan terekam kedalam sistem. Nilai bagi manajer • Ia perlu mengetahui laporan

	keuntungan dalam rentang waktu tertentu Nilai bagi bagian gudang • Ia perlu mengetahui produk apa saja yang berkurang
Apakah sistem pencatatan penjualan bergantung pada sesuatu?	Printer • Untuk mencetak struk Mesin debit ATM • Untuk menarik sejumlah uang pada *account* seseorang

Aktor



Dalam kasus ini, PELANGGAN tidak berinteraksi langsung dengan sistem.

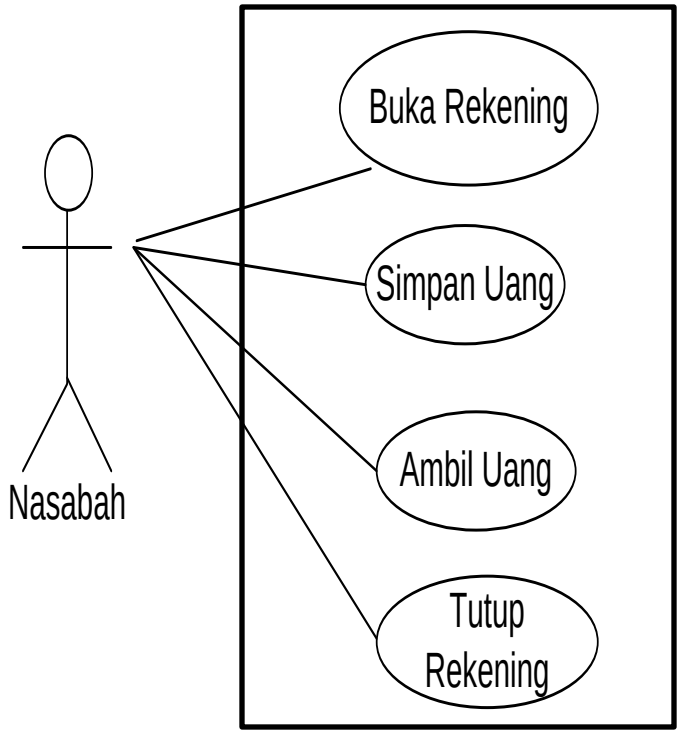
Use case

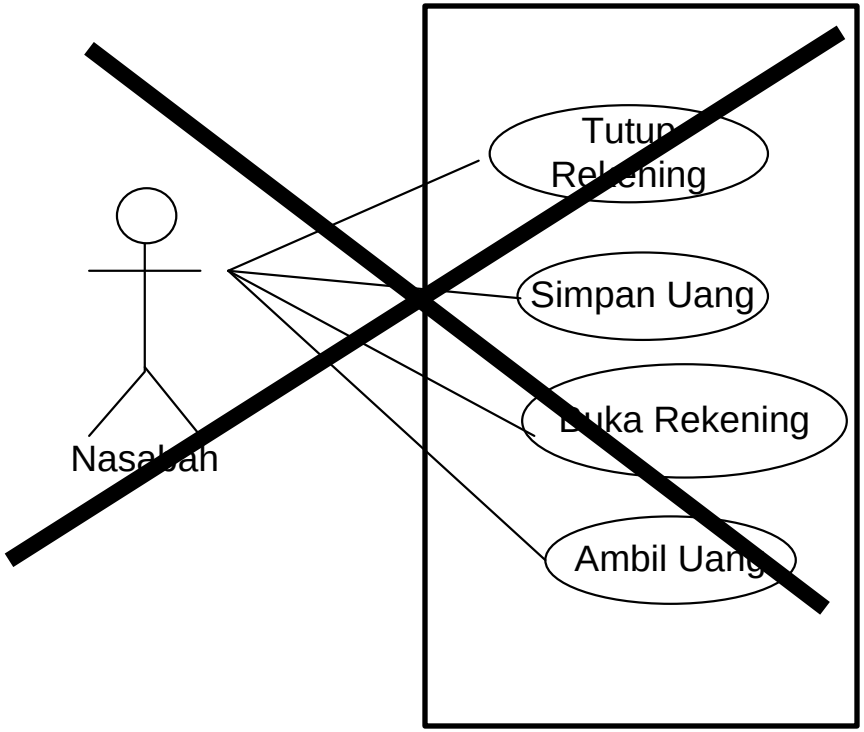
- Sebuah use case harus mendeskripsikan **sebuah pekerjaan dimana pekerjaan tersebut akan memberikan NILAI yang bermanfaat bagi aktor** (Kurt Bittner, Ian Spence. 2002).
- Untuk menemukan use cases, dimulai dari **sudut pandang aktor**, misalnya dengan bertanya:
  - Informasi apa sajakah yang akan didapatkan aktor dari sistem?
  - Apakah ada kejadian dari sistem yang perlu diberitahukan ke aktor?

Usecase

- Untuk menemukan use cases, dimulai dari **sudut pandang sistem**, misalnya dengan bertanya:
  - Apakah ada informasi yang perlu disimpan atau diambil dari sistem?
  - Apakah ada informasi yang harus dimasukkan oleh aktor?
- Use case diagram tidak terpengaruh urutan waktu, meskipun demikian supaya mudah dibaca perlu penyusunan use case

Usecase

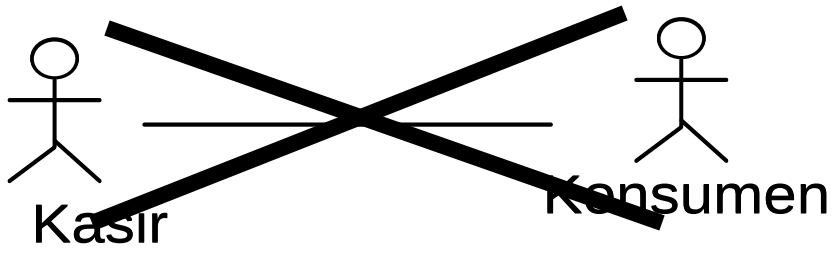

Buka Rekening
Simpan Uang
Ambil Uang
Tutup Rekening
Nasabah


Tutup Rekening
Simpan Uang
Buka Rekening
Ambil Uang
Nasabah

Aktor - Usecase

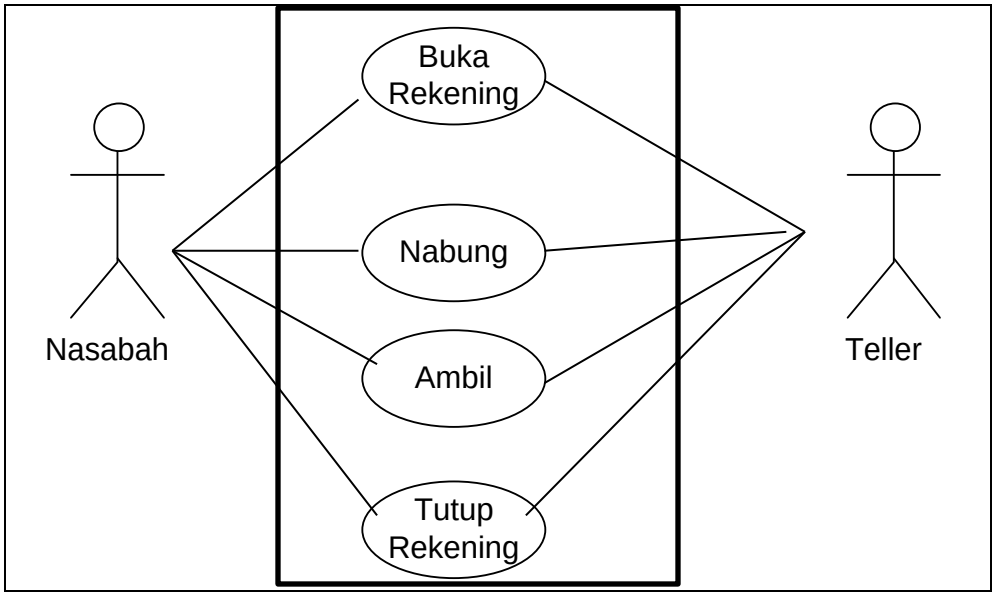
- Tidak boleh ada komunikasi langsung antar actor

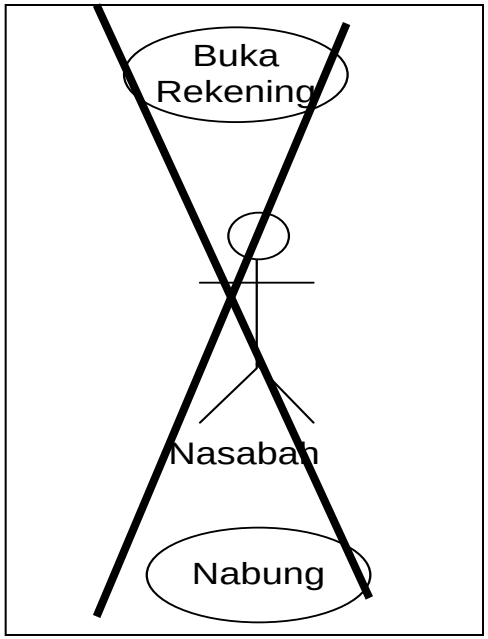
  (Actors don’t interact with one another)



Aktor - Usecase

- Letakkan actor utama pada pojok kiri atas dari diagram (people read from left to right, top to bottom)
- Actor tidak digambarkan ditengah-tengah use cases



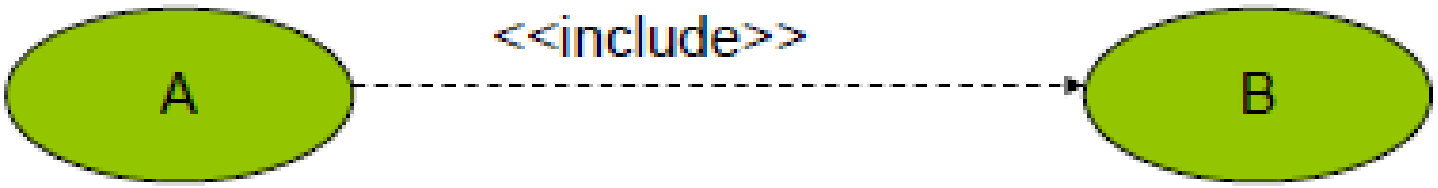


Relasi pada Use case

- Hubungan yang terjadi antar simbol dalam use case
- Menggunakan garis dan tipe simbol yang digunakan untuk menghubungkan garis.
- Ada 4 jenis relasi yang bisa timbul pada use case diagram:
  - Hubungan antara actor dan use case
  - Hubungan antara use case :
    - Include
    - Extend
  - Generalization/Inheritance antara use case
  - Generalization/Inheritance antara actors

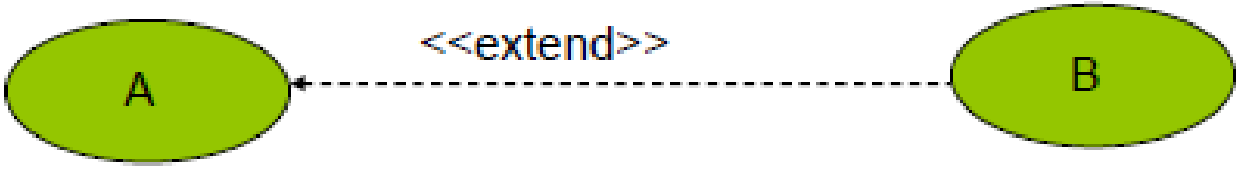
<<include>>

- Satu usecase dapat meng-include usecase lainnya
- Jika usecase A include ke usecase B, maka usecase B harus diimplementasi setiap kali usecase A dipanggil
- Digambarkan dengan garis putus-putus bertuliskan <<include>> ke arah usecase yang di-include



<<extend>>

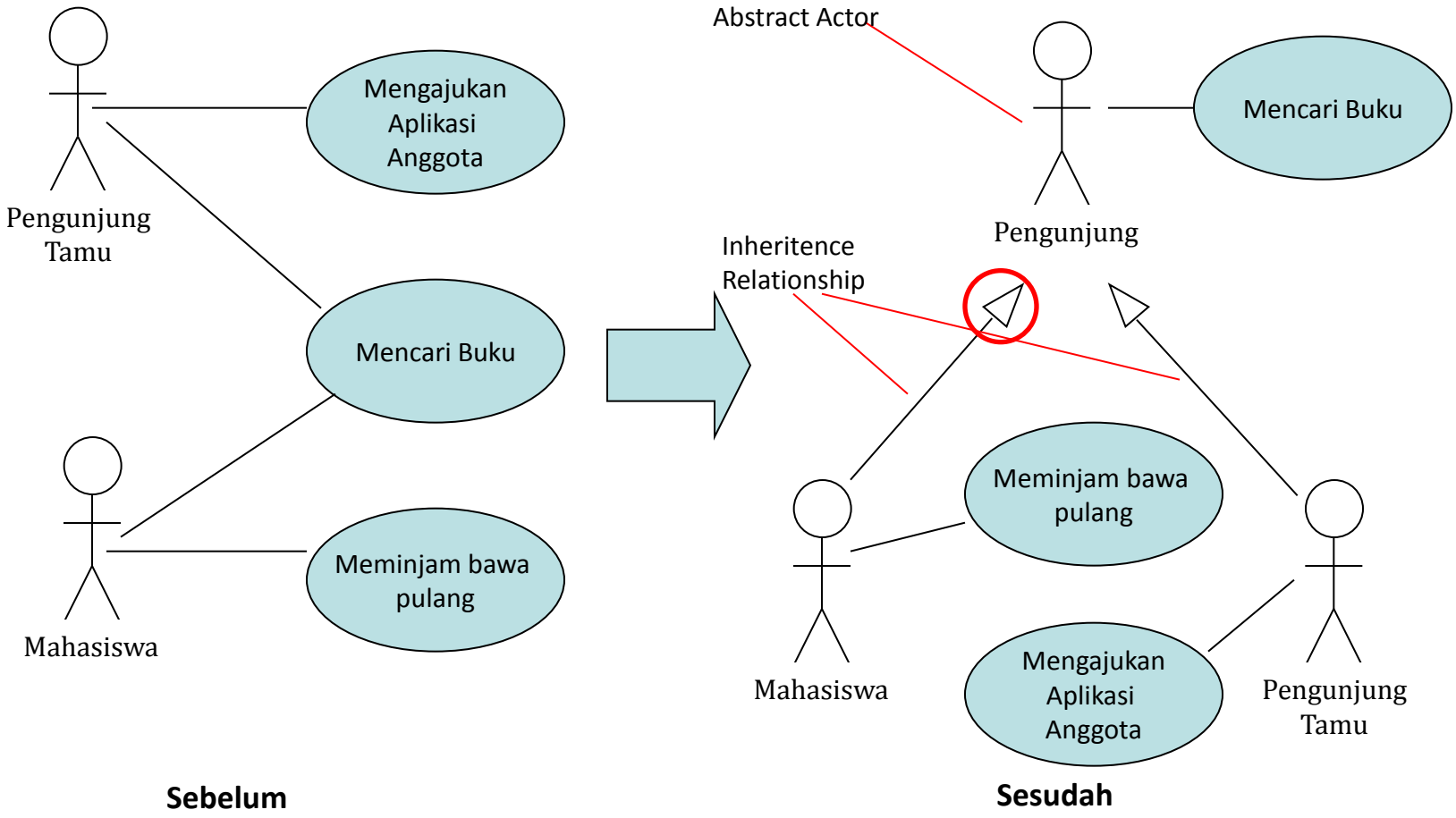
- Satu usecase dapat di-extend oleh usecase lain
- Jika usecase A di-extend oleh usecase B, maka antara usecase A dan B dapat diimplementasi secara bebas
- Digambarkan dengan garis putus-putus bertuliskan <<extend>> ke arah usecase yang di-extend
- Usecase A tidak harus memanggil usecase B dalam beberapa batasan



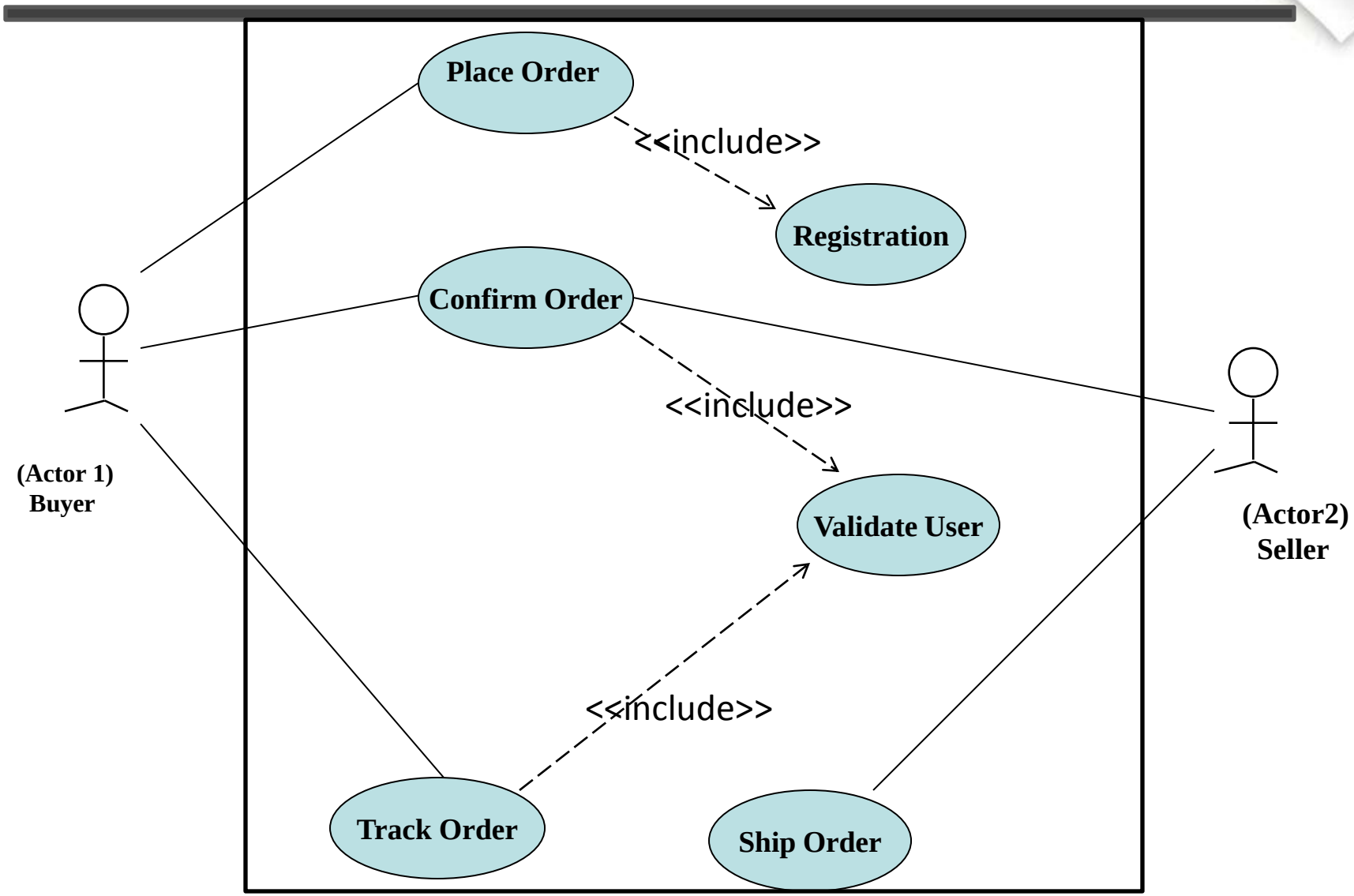
Generalization/ Inheritance

- Aktor dan usecase dapat dilakukan generalisasi
- Generalisasi digunakan untuk membuat aktor atau usecase yang lebih spesifik dari suatu aktor atau usecase
- Generalisasi aktor → digunakan pada saat dua atau lebih aktor berbagi behaviour yang umum, sehingga dibuat aktor baru untuk mengurangi komunikasi redudan dengan sistem

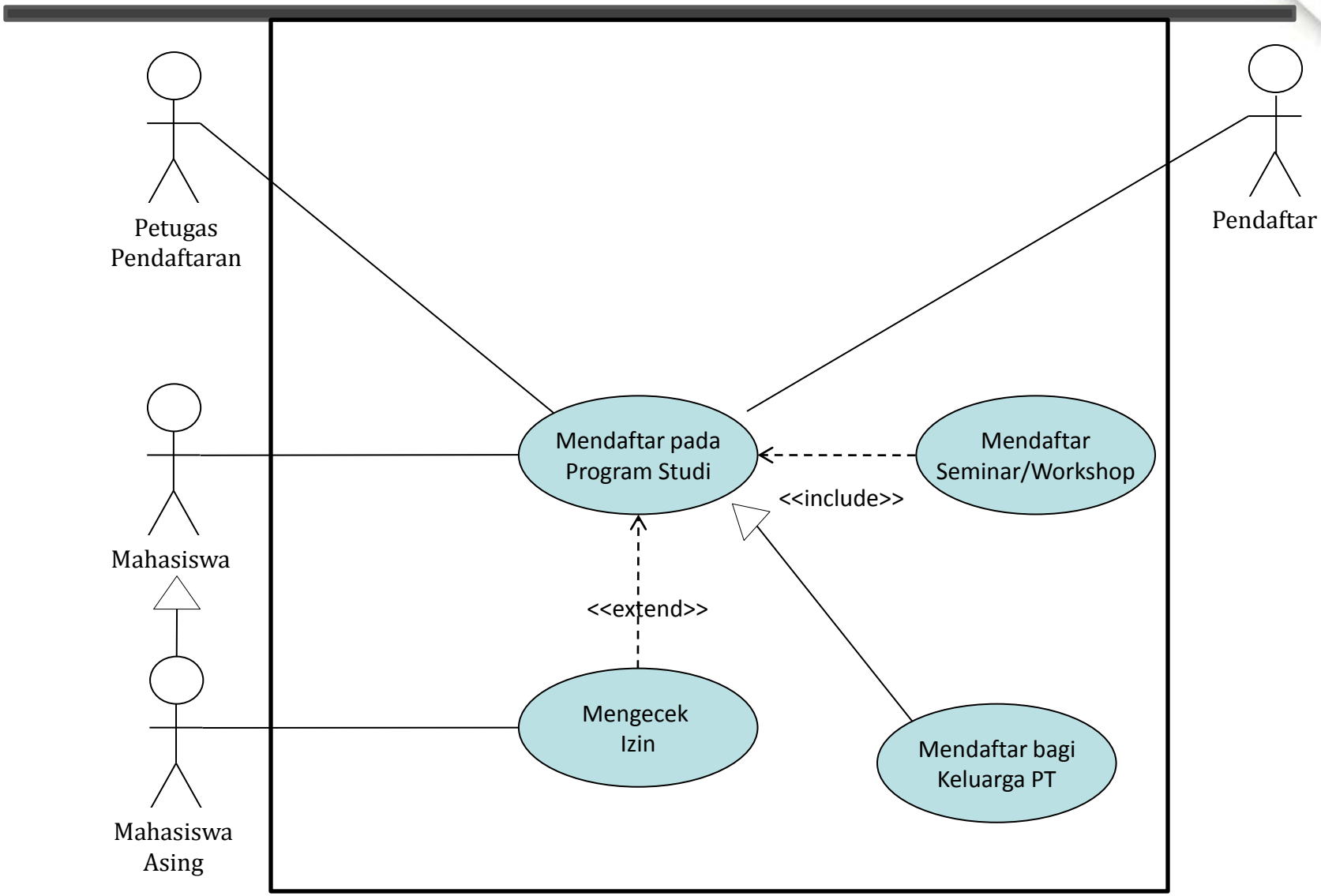
Generalization/ Inheritance



Contoh use case diagram



Contoh use case diagram



Use case scenario

- Use case scenario merupakan hasil inisiasi dari setiap use case
- Terbagi menjadi tiga bagian utama:
  - Identifikasi dan inisiasi use case
  - Tahapan aksi dan reaksi sistem
  - Kondisi dan asumsi

Use case scenario

Use case name	Accessing the HSVO tools and devices	
Related requirements	Email with information on how to log into the system, web address, username and password	
Goal in Context	Log into the HSVO system, and access the "Tools and Devices" section the system	
Preconditions	The user has the appropriate web address, username and password	
Successful End Condition	The user successfully logs in and is re-directed to the "Dashboard"	
Fail end condition	User is not able to log in	
Primary Actors	Student (or Tutor)	
Secondary Actors	None	
Trigger	The user's credentials are provided to the system for verification	
Main Flow	Step	Action
	1	User goes to the HSVO login page
	2	User enters details to login to the system: username, password, and selects current location
	3	The credentials are verified by the system
	4	The details are returned as verified and the user is redirected to personal dashboard. If there are any scheduled sessions comin up they are listed first; second, a list of recent activities or action within the account is presented, and third, a list of news and/or recommended tools is provided.
Extensions	Step	Branching action: Non verified credentials

Latihan

- Buatlah use case diagram untuk sistem informasi kepegawaian (sesuai kebutuhan dari hasil analisis)

Tugas Kelompok

- Buatlah sub bab analisis kebutuhan fungsional digambarkan dengan use case diagram
- Buatlah use case scenario nya lengkap untuk setiap use case nya.

Thank You