



**Kampus  
Merdeka**  
INDONESIA JAYA

MATA KULIAH – ANALISIS DAN PERANCANGAN SISTEM INFORMASI

## TOPIK 5: Object Modeling (Bagian Dua)

Disusun oleh:

Wahyu Indra Satria, S.Kom., M.Kom.



# *Learning Objectives*

(Tujuan Pembelajaran)

- ☐ Memahami *unified modeling language* (UML) dan beberapa contohnya
- ☐ Memahami dan merancang pemodelan *use case diagram* (UCD)



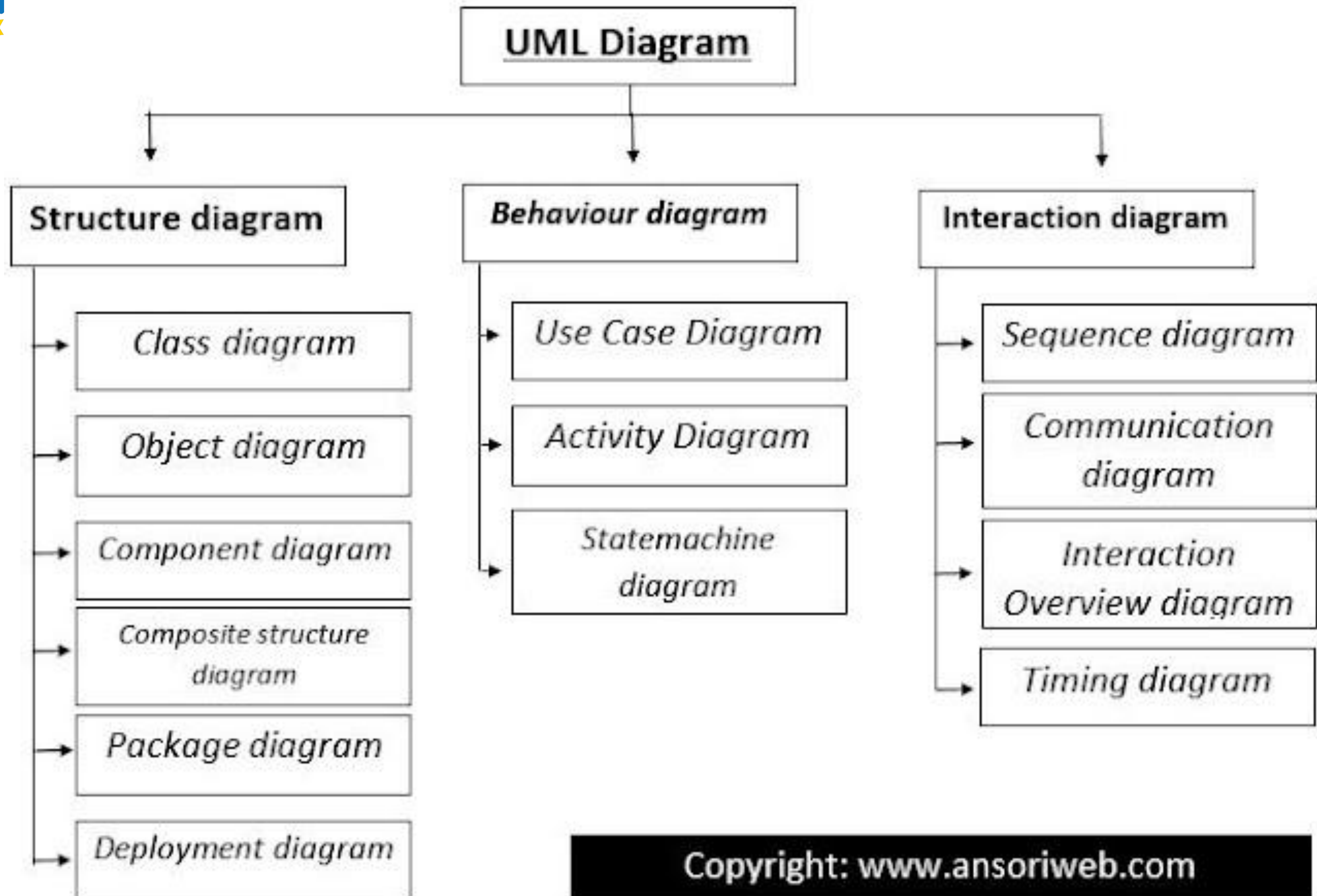
# Pengenalan *Unified Modeling Language* (UML)

## ► *Overview Unified Modeling Language (UML):*

- UML (*Unified Modeling Language*) adalah sekumpulan diagram yang digunakan untuk melakukan abstraksi (**pemodelan**) terhadap sebuah sistem atau perangkat lunak berbasis objek.
- UML dapat digunakan untuk mempermudah dalam proses pengembangan sistem/*software*/aplikasi yang berkelanjutan.
- UML dapat dikatakan juga sebagai perkembangan bahasa pemodelan di bidang rekayasa perangkat lunak, yang dimaksudkan untuk menyediakan cara standar dalam mem–visualisasikan desain sebuah sistem.
- UML terdiri dari banyak elemen–elemen grafis, yang kemudian digabungkan dalam suatu bentuk diagram.
- Tujuan representasi elemen–elemen grafis ke dalam diagram adalah untuk menyajikan beragam sudut pandang dari sebuah sistem, berdasarkan fungsi masing–masing diagram tersebut.
- Kumpulan dari beragam sudut pandang inilah yang disebut **sebuah model**.

Terkait materi topik ini, silahkan telusuri pada buku referensi 1 dimulai dari halaman 187.

# Pengenalan *Unified Modeling Language (UML)* (lanjutan...)

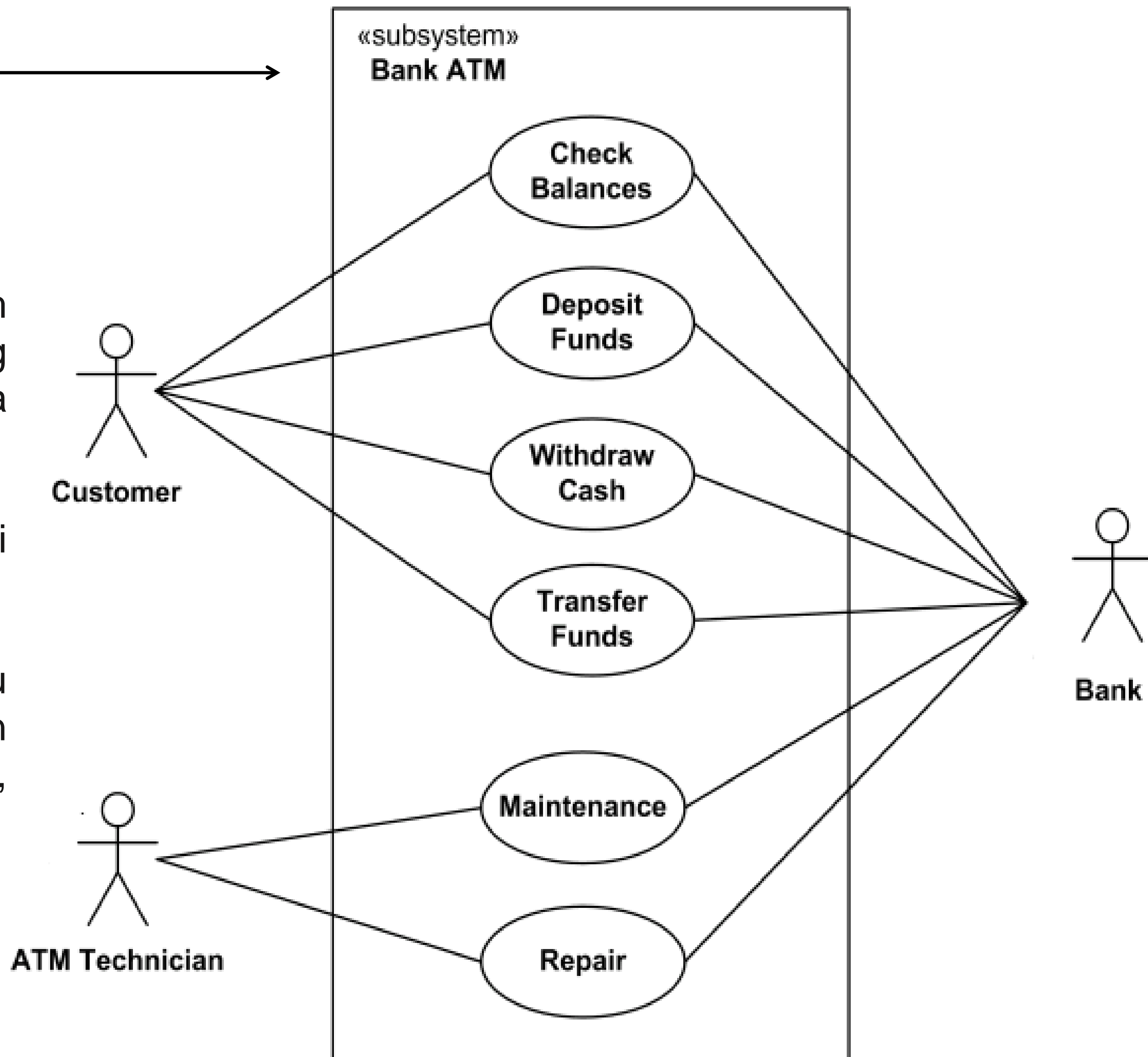




# Contoh–Contoh *Unified Modeling Language* (UML):

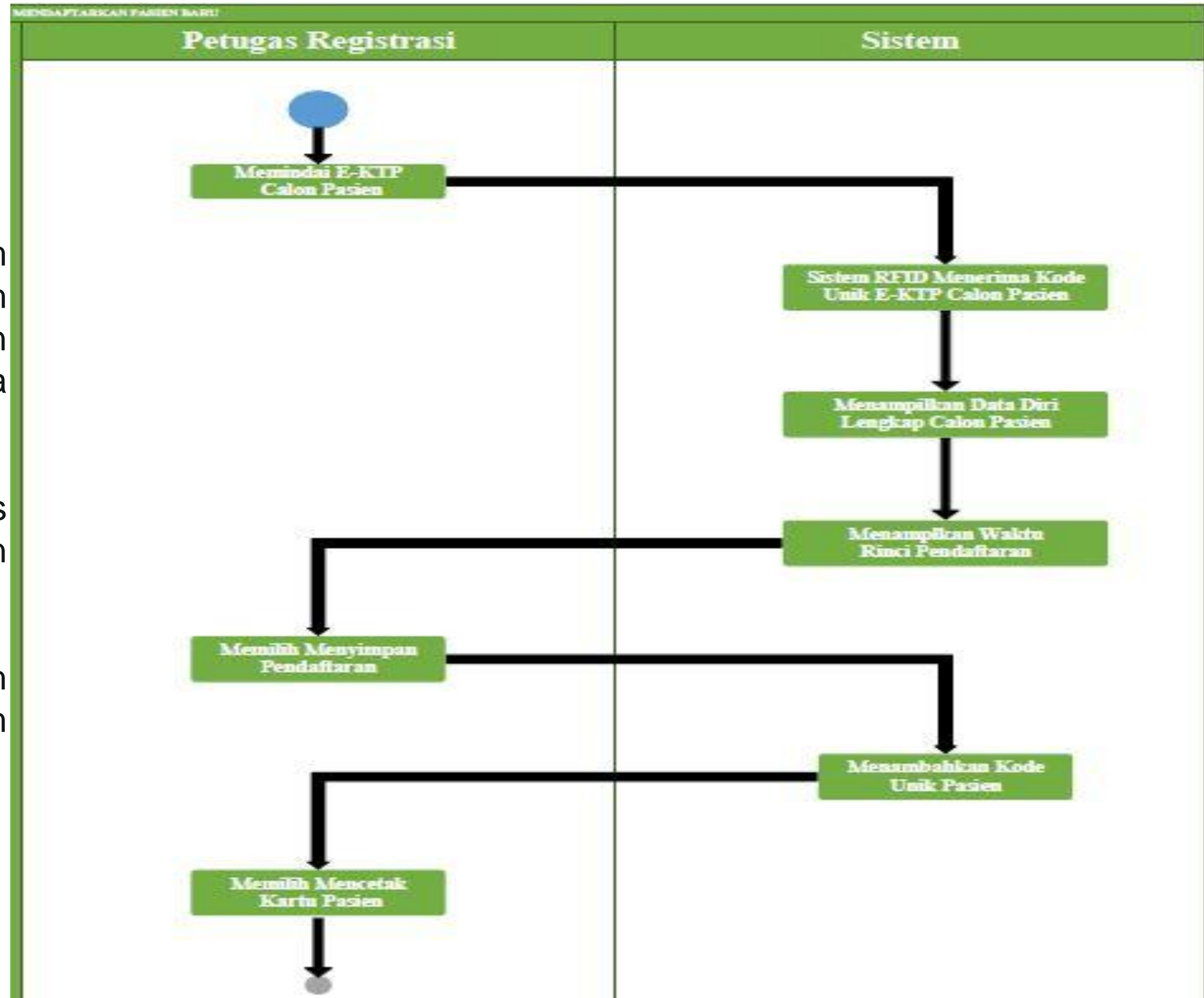
## ➤ *Use Case Diagram* (UCD):

- *Use Case Diagram* adalah satu jenis dari diagram UML (*Unified Modelling Language*) yang menggambarkan **hubungan interaksi** antara **sistem dan aktor**.
- *Use case* dapat mendeskripsikan tipe interaksi antara pengguna sistem dengan sistem itu sendiri.
- Langkah awal untuk melakukan pemodelan, perlu adanya suatu diagram yang mampu menjabarkan aksi aktor dengan aksi dalam sistem itu sendiri, seperti yang terdapat pada *Use Case Diagram*.



## ➤ *Activity Diagram*:

- Activity diagram atau diagram aktivitas, merupakan sebuah diagram yang dapat memodelkan berbagai proses yang terjadi pada suatu sistem.
- Seperti deskripsi runtutan proses berjalannya suatu sistem, dan digambarkannya secara vertikal.
- Activity diagram adalah salah satu contoh diagram UML dalam pengembangan dari Use Case.

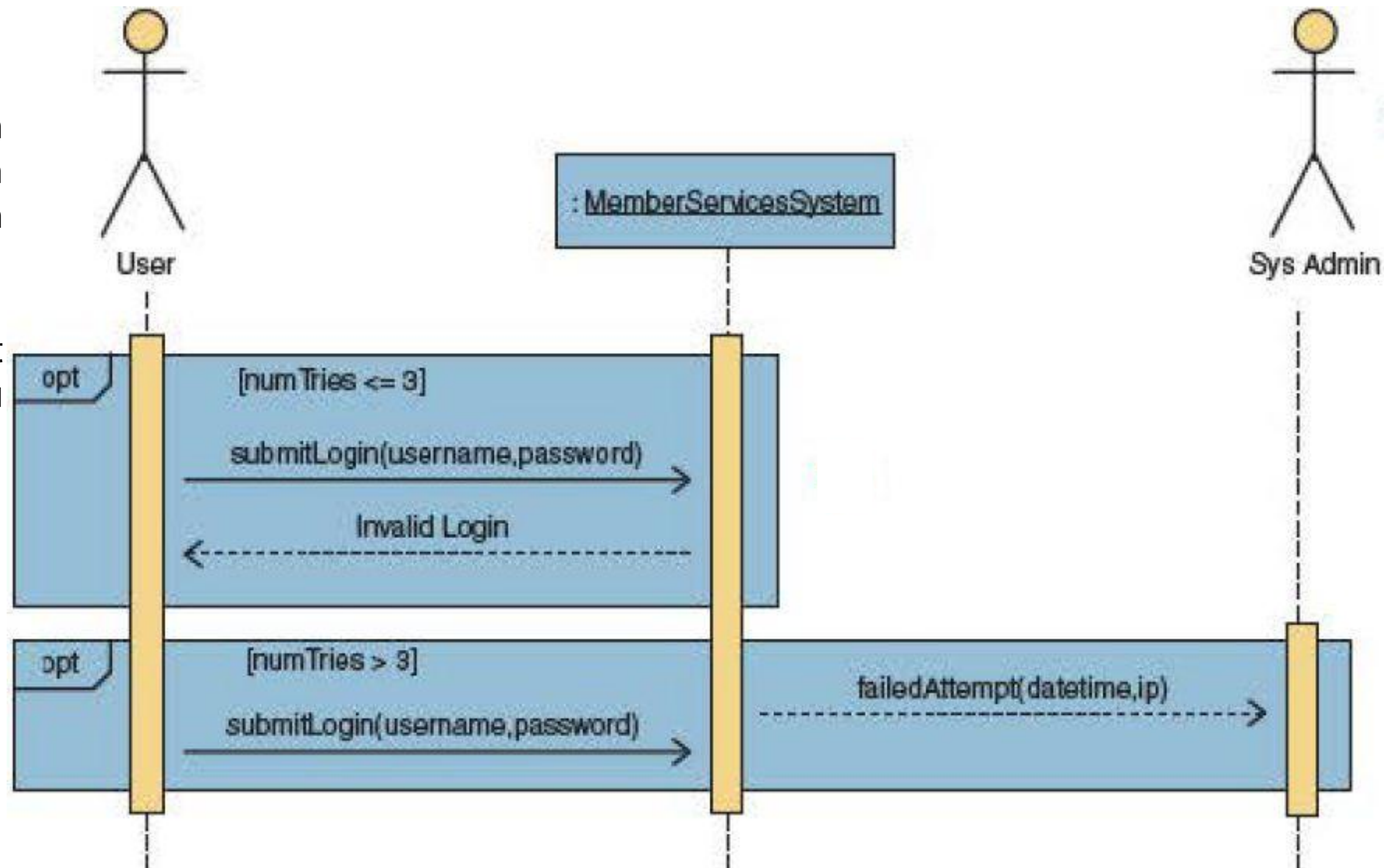




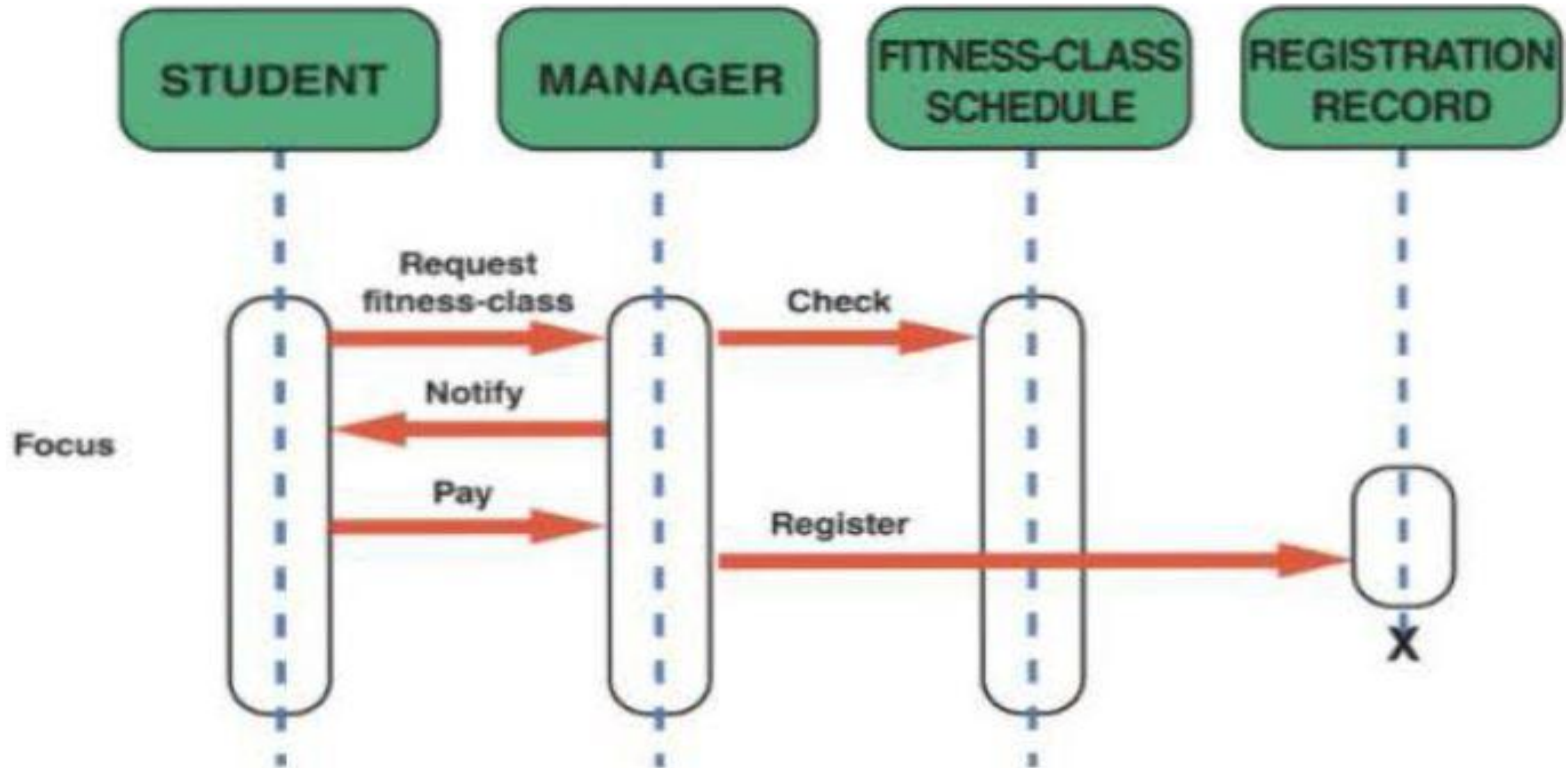
## System Sequence Diagram for Login Validation

### ➤ *Sequence Diagram*:

- Sequence diagram merupakan diagram yang menjelaskan interaksi objek berdasarkan urutan waktu.
- Sequence diagram dapat menggambarkan urutan atau tahapan yang harus dilakukan, agar menghasilkan sesuatu.







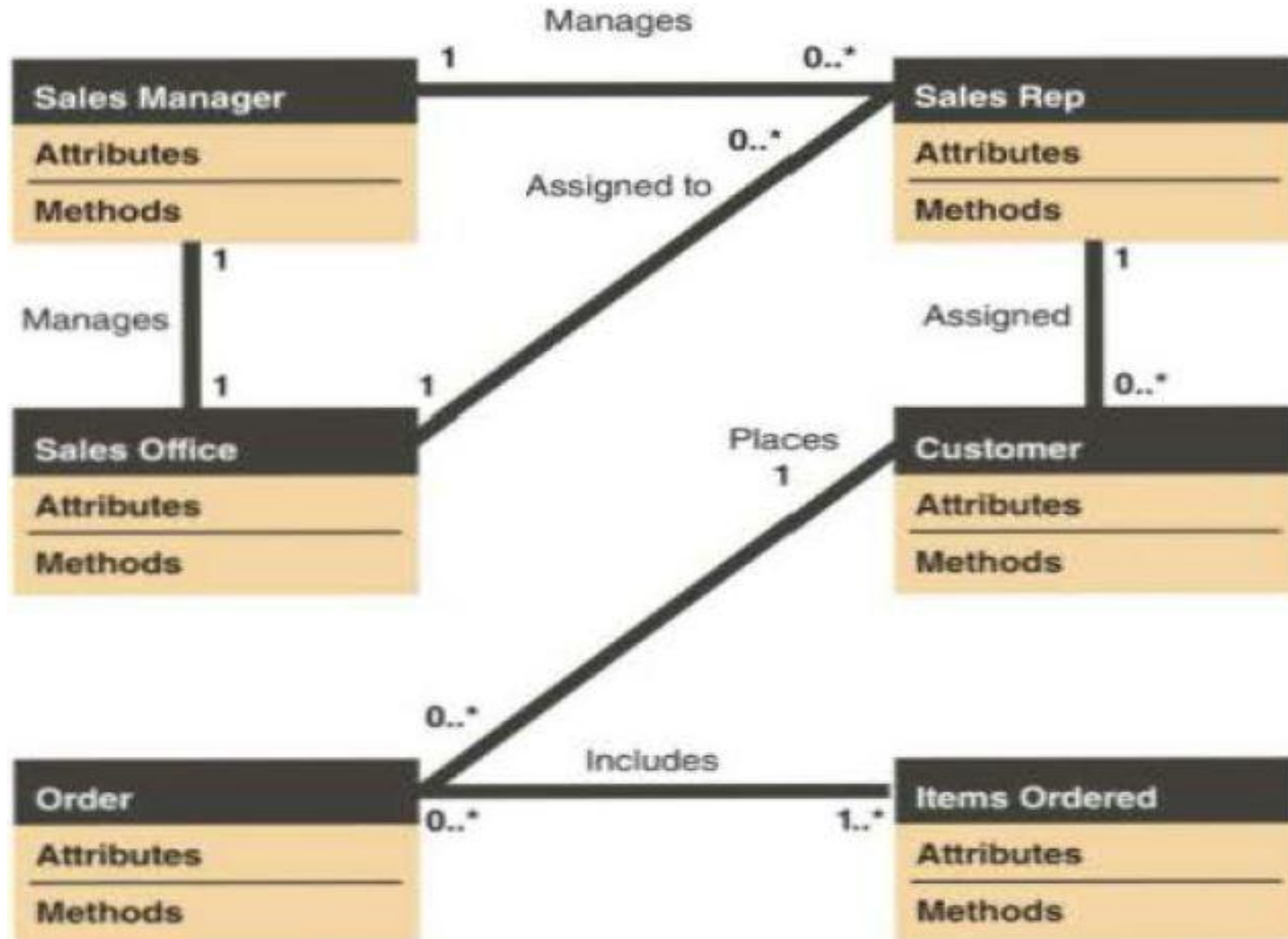
**FIGURE 6-19** The sequence diagram for the ADD NEW STUDENT use case. The use case description for ADD NEW STUDENT is shown in Figure 6-13.



### *Class Diagram*:

- *Class diagram* menunjukkan *class object* dan relasi yang terlibat dalam *use case*.
- Seperti *data flow diagram* (DFD), *class diagram* adalah *logical model*, yang berkembang menjadi *physical model* lalu menjadi sistem informasi yang berfungsi secara keseluruhan.
- Dalam analisis terstruktur, entitas, penyimpanan data, dan proses diubah menjadi struktur data beserta kode program.
- Demikian pula, *class diagram* berkembang menjadi *code module*, *data objects*, dan komponen sistem lainnya.
- Dalam *class diagram*, setiap *class* muncul sebagai persegi panjang, dengan nama *class* di bagian atas, diikuti oleh *class attributes* dan *class methods*.

## Class Diagram: (lanjutan...)



**FIGURE 6-17** Class diagram for a sales order use case (attributes and methods omitted for clarity).

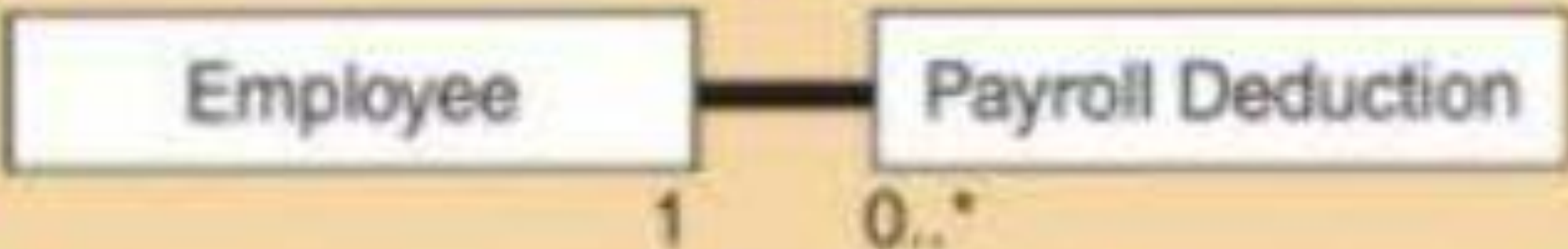
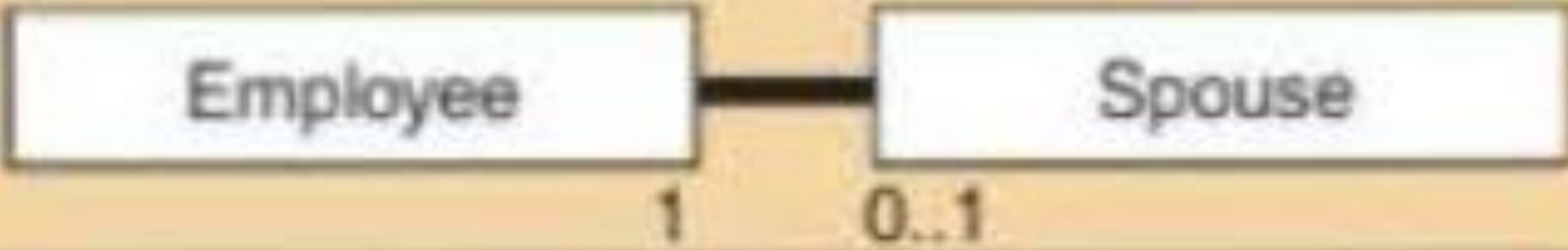
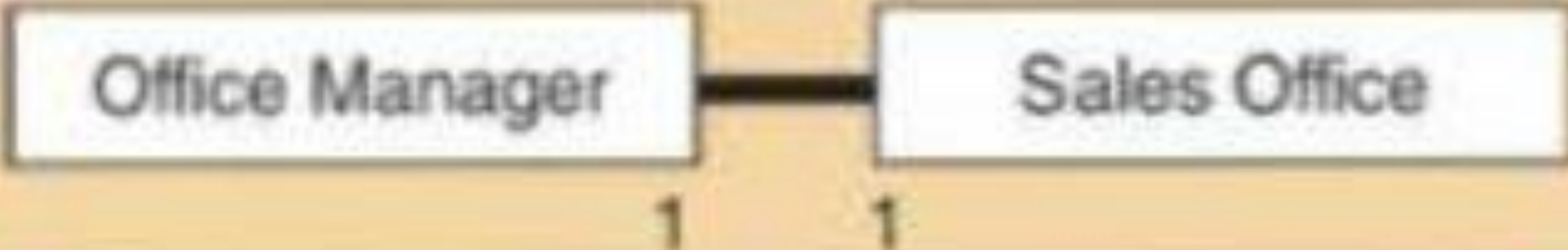



## Contoh–Contoh *Unified Modeling Language (UML)*: (lanjutan...)

### *Class Diagram*: (lanjutan...)

- Simbol garis menunjukkan hubungan antar *class*, dan memiliki label untuk mengidentifikasi tindakan yang menghubungkan kedua kelas.
- Untuk membuat *class diagram*, lakukan *review use case* dan identifikasi *class* dalam proses bisnis yang mendasarinya.
- *Class diagram* memiliki konsep yang disebut *cardinality*, yang menggambarkan bagaimana *instance* dari satu *class* berhubungan dengan *instance* dari *class* lain.
- Contohnya seorang karyawan mungkin tidak mendapatkan hari libur, satu hari libur, atau mungkin banyak hari libur.
- Demikian juga mungkin seorang karyawan tidak memiliki pasangan atau memiliki satu pasangan.



UML Notation	Nature of the Relationship	Example	Description
0..*	Zero or many		An employee can have no payroll deductions or many deductions.
0..1	Zero or one		An employee can have no spouse or one spouse.
1	One and only one		An office manager manages one and only one office.
1..*	One or many		One order can include one or many items ordered.

**FIGURE 6-16** Examples of UML notations that indicate the nature of the relationship between instances of one class and instances of another class.



# Memahami dan Merancang *Use Case Diagram*

## ➤ *Overview Use Case Diagram:*

- *Use case diagram* adalah satu jenis dari diagram UML (*Unified Modelling Language*) yang menggambarkan **hubungan interaksi** antara **sistem dan aktor**.
- *Use case* dapat mendeskripsikan tipe interaksi antara pengguna sistem dengan sistem itu sendiri.
- Langkah awal untuk melakukan pemodelan, perlu adanya suatu diagram yang mampu menjabarkan aksi aktor dengan aksi dalam sistem itu sendiri, seperti yang terdapat pada *use case diagram*.
- *Use case diagram* adalah ringkasan visual dari beberapa *use case* terkait dalam suatu sistem atau subsistem.
- Sebagai contoh departemen *service* mobil yang khas, seperti yang ditunjukkan pada **FIGURE 6–14** (pada slide 14) .
- Departemen *service* melibatkan pelanggan, pencatat *service* untuk menyiapkan perintah kerja dan faktur, serta mekanik yang melakukan pekerjaan perawatan/perbaikan mobil.





**FIGURE 6-14** A typical auto service department might involve customers, service writers who prepare work orders and invoices, and mechanics who perform the work.



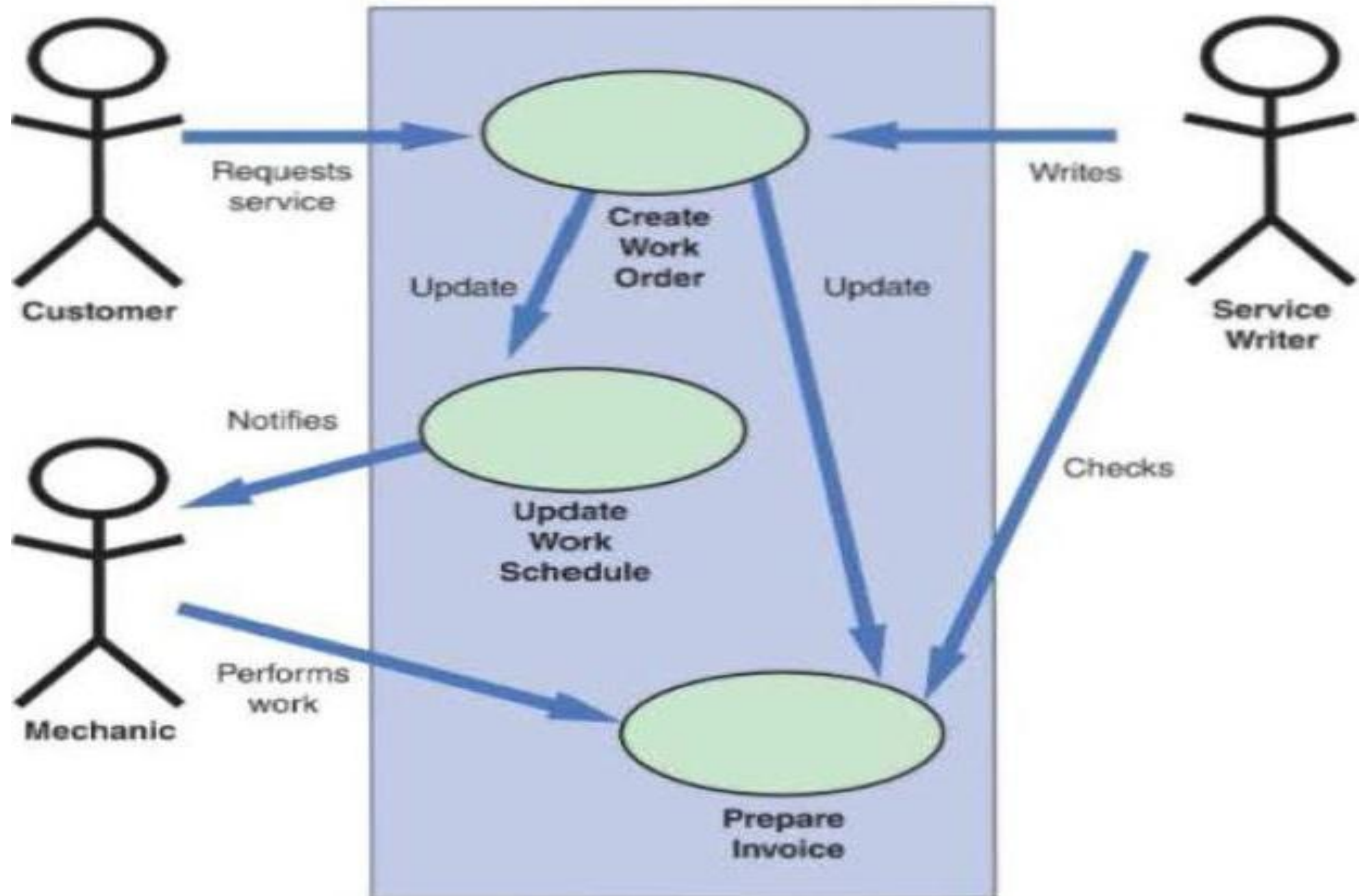
# Memahami dan Merancang *Use Case Diagram* (lanjutan...)

## Overview *Use Case Diagram*: (lanjutan...)

- Ketika *use case diagram* dibuat, langkah pertama adalah mengidentifikasi batas sistem, yang diwakili oleh sebuah persegi panjang.
- Batas sistem menunjukkan apa yang termasuk dalam sistem (di dalam persegi panjang) dan apa yang tidak termasuk dalam sistem (di luar persegi panjang).
- Setelah batas sistem diidentifikasi, *use case* ditempatkan pada diagram, aktor ditambahkan, lalu *relationship*-nya ditampilkan.
- Contohnya pada **FIGURE 6–15** (pada slide 16).

# Memahami dan Merancang *Use Case Diagram* (lanjutan...)

Use Case Diagram: Auto Service Department



**FIGURE 6-15** A use case diagram to handle work at an auto service department.





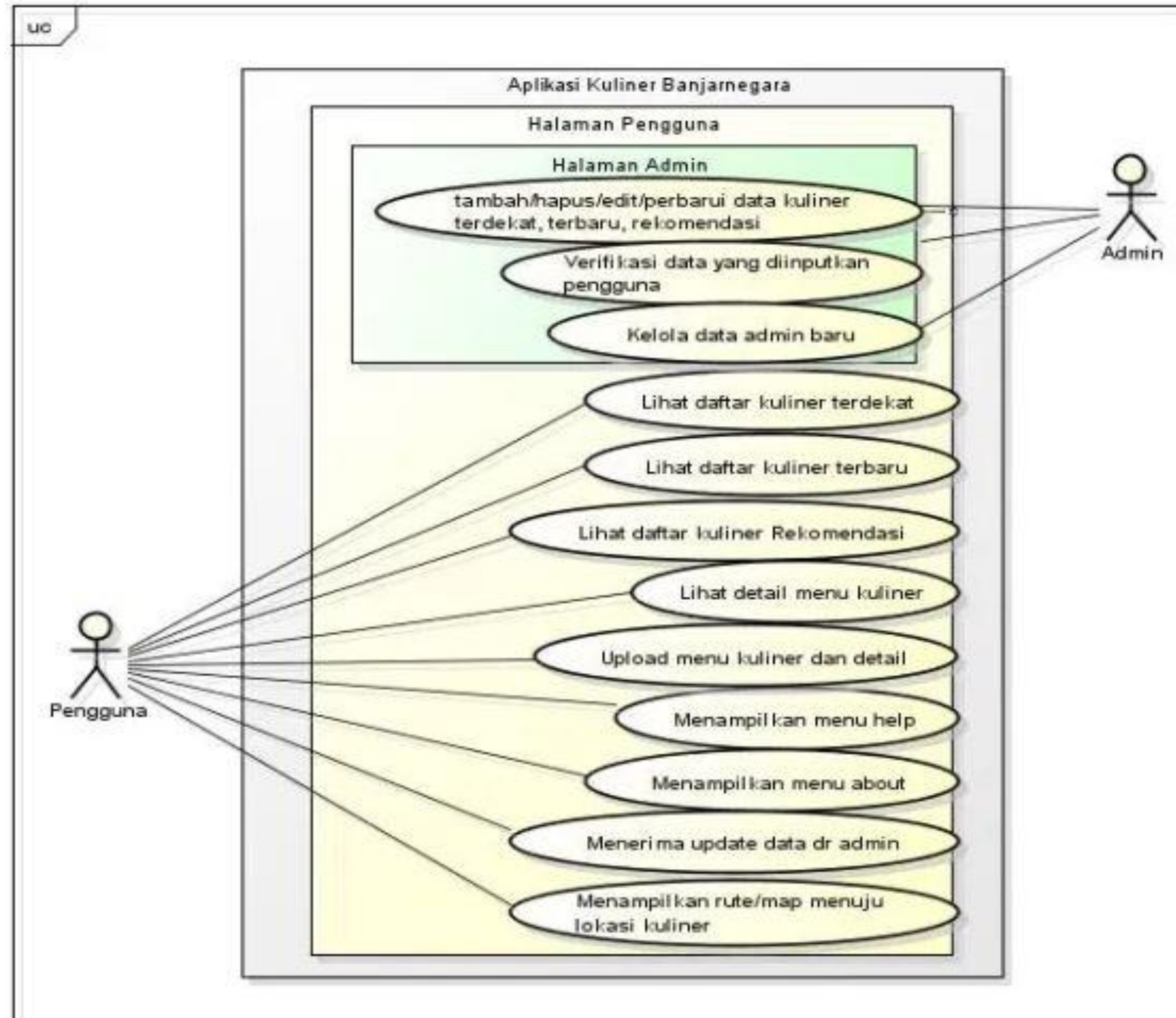
# Memahami dan Merancang *Use Case Diagram* (lanjutan...)

## ➤ *Overview Use Case Diagram:* (lanjutan...)

- *Use case diagram* adalah gambaran urutan interaksi yang saling berkaitan antara suatu sistem dengan aktor yang terlibat.
- *Use case diagram* menggambarkan tipe interaksi antara pengguna/*user* suatu program/sistem dengan sistem itu sendiri.
- *Use case* menggambarkan bagaimana sebuah sistem digunakan, serta dipakai juga untuk membentuk perilaku (*behaviour*) sistem yang akan dibuat.
- *Use case* menggambarkan interaksi antara pengguna (aktor) dengan sistem yang sudah ada, serta memaparkan proses kegiatan dan proses bisnis yang dilakukan oleh pengguna (aktor).
- Pada *use case diagram*, setiap proses digambarkan kedalam sebuah *use case* berbentuk oval dengan relasi/hubungan yang ada didalamnya.



➤ **Contoh UCD:**  
Penjelasannya ada  
pada slide 19.















# Memahami dan Merancang *Use Case Diagram* (lanjutan...)

➤ **Contoh UCD:**  
**Penjelasan UCD** yang  
ada pada slide 18.

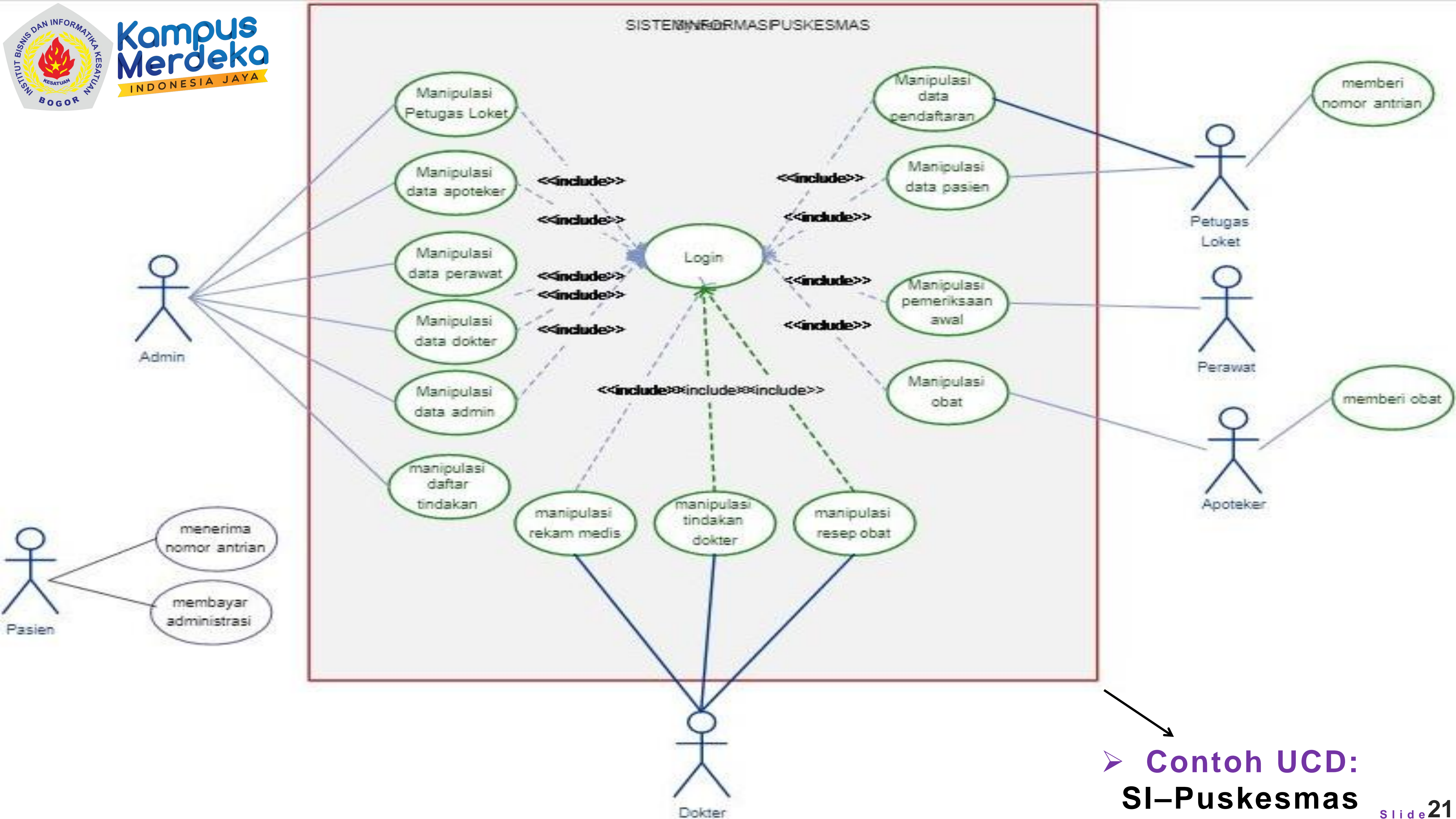
No	User	Deskripsi
1	Menampilkan wisata kuliner terdekat	Sistem dapat menampilkan list wisata kuliner yang terdekat dari lokasi user dan detail kuliner.
2	Menampilkan wisata kuliner terbaru	Sistem dapat menampilkan list wisata kuliner yang terbaru beserta detail kuliner, seperti harga dan rumah makan wisata kuliner yang terbaru.
3	Menampilkan wisata kuliner rekomendasi	Sistem akan menampilkan list wisata kuliner rekomendasi yang diperoleh dari feedback.
4	Menampilkan detail menu wisata kuliner	Sistem akan menampilkan detail informasi mengenai menu wisata kuliner tersebut. Seperti nama kuliner, harga, feedback, jarak, dan gambar kuliner.
5	About	Sistem akan menampilkan tentang pembuat aplikasi.
6	Help	Sistem menampilkan semua update dari admin seperti menu kuliner yang terbaru.
7	Dapat menerima update dari admin	Sistem akan menampilkan semua update dari admin, seperti menu kuliner yang terbaru.
8	Upload menu kuliner	Pengguna dapat mengupload sendiri menu yang ingin ditampilkan di aplikasi.
9	Menampilkan rute menuju lokasi wisata kuliner	Sistem akan menampilkan peta navigasi/rute dari lokasi user sekarang menuju lokasi wisata kuliner.



➤ **Contoh UCD:**  
**Simbol-simbol UCD.**

GAMBAR	NAMA	KETERANGAN
	<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
	<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri ( <i>independent</i> ) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
	<i>Generalization</i>	Hubungan dimana objek anak ( <i>descendent</i> ) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk ( <i>ancestor</i> ).
	<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara <i>eksplisit</i> .
	<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
	<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.
	<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
	<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
	<i>Collaboration</i>	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya ( <i>sinergi</i> ).
	<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi.







## ➤ Tujuan, Fungsi, dan Manfaat *Use Case Diagram*:

**Tujuan dari *use case diagram* antara lain:**

- Memetakan kebutuhan sistem.
- Merepresentasikan interaksi pengguna terhadap sistem.
- Untuk mengetahui kebutuhan diluar sistem.

**Adapun fungsi dari *use case diagram* sebagai berikut:**

- Dapat menggambarkan urutan aktivitas proses yang ada pada suatu sistem.
- Dapat menggambarkan proses bisnis dan juga urutan aktivitas yang ada dalam sebuah proses.

**Manfaat *use case diagram* adalah:**

- Memudahkan hubungan dengan menggunakan *domain expert* dan juga *end user*.
- Adanya *interface* yang harus dimiliki oleh sebuah sistem.
- Memberikan kepastian pemahaman yang pas, tentang *requirement* atau kebutuhan pada sebuah sistem.
- Dapat digunakan untuk mengidentifikasi, siapa yang sedang berinteraksi dengan sistem, dan juga apa yang harus dilakukan untuk sistem tersebut.
- Biasanya digunakan untuk melakukan verifikasi sistem yang dikembangkan.



## ➤ Karakteristik *Use Case Diagram*:

Adapun ciri–ciri dari *use case diagram* antara lain:

- *Use case* merupakan interaksi atau dialog antara sistem dan pengguna (*actor*), termasuk peralihan pesan serta tindakan yang dilakukan oleh suatu sistem.
- *Use case* diprakarsai oleh pengguna/*user* (*actor*) suatu sistem dan mungkin melibatkan peran *actor* yang lain.
- *Use case* harus menyediakan nilai minimal kepada satu pengguna (*actor*).
- *Use case* bisa mempunyai perluasan yang menjelaskan tindakan khusus dalam interaksi, atau mungkin juga menyisipkan *use case* lain.
- *Use case class* mempunyai objek yang disebut skenario.
- Skenario menyatakan urutan pesan dan reaksi tunggal.

## REFERENSI

- ❑ Tilley, Scott, System Analysis and Design, CENGAGE, USA, 2020.
- ❑ Dennis, Alan, System Analysis and Design: An Object Oriented Approach with UML, Wiley, USA, 2015.
- ❑ Satzinger, Jackson, Burd, System Analysis and Design in A Changing World, CENGANE, USA, 2012.
- ❑ Langer, Arthur, Analysis and Design of Information Systems, Springer, USA, 2008.
- ❑ Bentlet, Whitten, System Analysis and Design Methods, McGraw–Hill Irwin, USA, 2007.
- ❑ Wasson, Charles, System Analysis, Design, and Development, Concepts, Principles, and Practices, Wiley–Interscience, Canada, 2006.
- ❑ researchgate.com
- ❑ salaryexplorer.com.
- ❑ <https://www.gotutorid.com/java/class-dan-object/>
- ❑ [https:// www.ansoriweb.com](https://www.ansoriweb.com)





TERIMA KASIH