

## Penerapan UML Class Diagram dalam Perancangan Sistem Informasi Perpustakaan Berbasis Web

Sandy Wahyu Ramdany\*<sup>1</sup>, Sarah Aulia Kaidar<sup>2</sup>, Banly Aguchino<sup>3</sup>, Chika Amelia Alira Putri<sup>4</sup>,  
Ruiz Anggie<sup>5</sup>

<sup>1,2</sup> Teknik Industri, Fakultas Teknik, Universitas Bhayangkara Jakarta Raya, Bekasi Indonesia

e-mail: [1202110215108@mhs.ubharajaya.ac.id](mailto:1202110215108@mhs.ubharajaya.ac.id), [2202110215116@mhs.ubharajaya.ac.id](mailto:2202110215116@mhs.ubharajaya.ac.id),

[3202110215109@mhs.ubharajaya.ac.id](mailto:3202110215109@mhs.ubharajaya.ac.id), [4202110215107@mhs.ubharajaya.ac.id](mailto:4202110215107@mhs.ubharajaya.ac.id),

[5202110215111@mhs.ubharajaya.ac.id](mailto:5202110215111@mhs.ubharajaya.ac.id)

\* Korespondensi: [paduloh@dsn.ubharajaya.ac.id](mailto:paduloh@dsn.ubharajaya.ac.id)

### ABSTRACT

*The library is an important component of learning resources that every school and college must have. Apart from that, libraries also act as information centers, sources of knowledge, and places to conduct research. In the library, there are various processes such as borrowing books, returning books, extending the loan period, and procuring new books. Currently, the book return process at the library uses a barcode system to input data on books borrowed. However, there are often long queues when returning books, limited library operating hours, and the involvement of library staff in the book return process add to the obstacles. Therefore, we need a system that allows returning books without the help of library staff and does not depend on library operating hours. Software has been developed to verify book returns, provide information about book returns, as well as notify books that have not been returned and the total fines that must be paid by library members.*

**Keywords:** Libraries, Information centers, Research, Queues, Learning resources

### ABSTRAK

Perpustakaan merupakan komponen penting dari sumber belajar yang wajib dimiliki oleh setiap sekolah dan perguruan tinggi. Selain itu, perpustakaan juga berperan sebagai pusat informasi, sumber ilmu pengetahuan, dan tempat untuk melakukan penelitian. Di perpustakaan, terdapat berbagai proses seperti peminjaman buku, pengembalian buku, perpanjangan masa pinjam, dan pengadaan buku baru. Saat ini, proses pengembalian buku di perpustakaan menggunakan sistem barcode untuk menginput data buku yang dipinjam. Namun, seringkali terjadi antrian panjang saat pengembalian buku, jam operasional perpustakaan yang terbatas, serta keterlibatan petugas perpustakaan dalam proses pengembalian buku menambah kendala. Oleh karena itu, dibutuhkan sebuah sistem yang memungkinkan pengembalian buku tanpa bantuan petugas perpustakaan dan tidak tergantung pada jam operasional perpustakaan. Perangkat lunak telah dikembangkan untuk memverifikasi pengembalian buku, memberikan informasi tentang pengembalian buku, serta memberitahukan buku yang belum dikembalikan dan total denda yang harus dibayar oleh anggota perpustakaan.

**Kata Kunci:** Perpustakaan, Pusat informasi, Penelitian, Antrian, Sumber belajar

### PENDAHULUAN

Perpustakaan merupakan bagian dari sumber belajar yang harus dimiliki oleh setiap sekolah atau perguruan tinggi. Perpustakaan juga merupakan salah satu pusat informasi, sumber ilmu pengetahuan dan penelitian.

Perpustakaan menjadi tempat sumber informasi sehingga dapat dinikmati oleh banyak orang. Pada perpustakaan memiliki beberapa proses yang dilakukan yaitu proses peminjaman buku, proses pengembalian buku, proses perpanjangan buku, serta proses pengadaan

buku. Dalam proses pengembalian buku yang terjadi di perpustakaan saat ini sudah menggunakan sebuah sistem *barcode* yang digunakan dalam menginputkan data buku yang dipinjam. Proses pengembalian buku di perpustakaan saat ini seringkali terjadi antrian pada saat pengembalian buku, jam operasional dari perpustakaan yang terbatas serta diperlukannya petugas perpustakaan untuk proses pengembalian buku. Oleh karena itu dibutuhkan sebuah sistem yang tidak memerlukan bantuan dari petugas perpustakaan untuk mengembalikan buku dari para pengguna perpustakaan, proses pengembalian buku dapat dilakukan tanpa melihat jam operasional dari perpustakaan. Untuk itu dibuatlah sebuah perangkat lunak yang mampu melakukan verifikasi pengembalian buku dan dapat memberikan informasi pengembalian buku. Serta dapat memberikan informasi buku yang belum dikembalikan oleh anggota perpustakaan dan total keseluruhan denda yang harus dibayar oleh anggota perpustakaan.

Perancangan sistem informasi yang terstruktur dan sistematis merupakan langkah penting dalam pengembangan sistem informasi yang berkualitas. UML (*Unified Modeling Language*) merupakan bahasa pemodelan yang banyak digunakan untuk merancang sistem perangkat lunak. UML menyediakan berbagai diagram untuk memodelkan aspek-aspek sistem, termasuk *Class Diagram*.

*Class Diagram* digunakan untuk memodelkan kelas-kelas dalam sistem, atributnya, metodenya, dan hubungan antar kelas. Penerapan *Class Diagram* dalam perancangan sistem informasi perpustakaan diharapkan dapat menghasilkan model sistem yang jelas, terstruktur, dan mudah dipahami. Model sistem yang baik akan menjadi dasar yang kuat untuk pengembangan dan implementasi sistem informasi perpustakaan yang efektif.

## Sistem

Sistem merupakan suatu tatanan yang terdiri dari sejumlah komponen fungsional dengan tugas atau fungsi khusus yang berkaitan dan kemudian secara bersama – sama memiliki tujuan untuk memenuhi suatu proses atau pekerjaan tertentu (Alzedan dalam Gede Endra Bratha, 2022).

Sistem merupakan suatu bentuk integrasi antara satu komponen dengan komponen lainnya, karena sistem memiliki sasaran yang berbeda untuk setiap kasus yang terjadi yang ada didalam sistem tersebut (Fitriyana & Sucipto, 2020).

## Informasi

Informasi merupakan data yang telah diproses atau memiliki arti. Adapun karakteristik penting yang harus dimiliki oleh informasi, seperti: relevansi, akurat, ketepatan waktu, dan kelengkapan.

## Sistem Informasi

Sistem informasi manajemen merupakan suatu metode yang disusun guna memberikan suatu informasi yang tepat waktu untuk manajemen yang berkaitan dengan lingkungan di luar organisasi dan juga kegiatan operasi di dalam suatu organisasi yang bertujuan untuk memberikan suatu kemudahan bagi proses manajemen dan memperbaiki proses perencanaan dan pengawasan serta menunjang proses pengambilan keputusan (Gede Endra Bratha, 2022). Pada dasarnya sistem informasi dapat dibedakan menjadi dua, yaitu sistem yang terstruktur (*formal*) dan sistem yang tidak terstruktur (*nonformal*). Sistem *formal* merupakan suatu sistem yang dijalankan sesuai dengan norma – norma organisasi yang berlaku untuk semua kalangan masyarakat, sesuai dengan kedudukannya dalam sebuah organisasi. Sistem *formal* ini bergantung pada tugas, wewenang, serta tanggung jawab yang telah dibebankan kepada pejabat organisasi. Sedangkan sistem *nonformal* merupakan sistem yang berlaku pada suatu lingkungan organisasi melalui saluran – saluran yang tidak resmi, akan tetapi masih mempunyai pengaruh yang cukup kuat dalam kehidupan organisasi tersebut.

Sistem informasi juga merupakan suatu sistem dalam suatu organisasi yang merupakan kombinasi dari orang – orang, fasilitas, teknologi, media dan pengendalian yang ditujukan untuk mendapatkan jalur komunikasi penting, memproses tipe transaksi rutin tertentu, memberi sinyal kepada manajemen dan yang lainnya terhadap kejadian-kejadian *internal* dan *external* yang penting serta menyediakan suatu dasar informasi untuk pengambilan keputusan yang tepat (Elisabet

Yunaeti Anggraeni dalam Sandfreni et al., 2021).

### Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat-sifat tertentu, karakteristik sistem atau sifat-sifat tertentu itu adalah sebagai berikut:

- a. **Komponen**  
Komponen mutlak diperlukan karena merupakan sub sistem daripada sistem.
- b. **Batasan *system***  
Sistem yang dibangun perlu ada batasan yang jelas supaya tujuan dari sistem dapat tercapai. Bila batasan sistem tidak jelas maka tujuan sistem akan tidak jelas dan tidak sesuai dengan target yang diinginkan.
- c. **Lingkungan di luar dan di dalam *system***  
Lingkungan sistem dibangun, bila tidak dijaga bisa mempengaruhi sistem.
- d. **Antar muka**  
Antar muka diperlukan untuk menghubungkan sistem dengan sub sistem penentukunya.
- e. ***Input***  
Data mentah yang sudah didapat diinputkan kedalam penyimpanan data yang sudah disiapkan. *Input* data diperlukan karena bisa saja data mentah yang diperoleh data cetak atau tulisan tangan, sehingga perlu *diinputkan* melalui komputer.
- f. ***Output***  
Suatu sistem tidak bisa dikatakan dibuat bila tidak ada hasil baik berupa file atau cetakan yang diharapkan.
- g. **Proses**  
Suatu sistem bisa dikatakan telah melakukan aktifitasnya bila terjadi proses yang mengubah input menjadi *output* yang diharapkan.
- h. **Tujuan**  
Sistem tanpa tujuan yang pasti akan sia-sia. Berdasarkan uraian diatas karakteristik sistem atau kasus memiliki unsur, media dan misi (tujuan), karena sistem adalah gabungan dari beberapa komponen yang paling berbeda tetapi saling ketergantungan.

### StarUML

StarUML merupakan proyek *open source* untuk mengembangkan *platform Unified Modeling Language (UML)* atau *Model Driven Architecture (MDA)* yang cepat, fleksibel, dapat diperluas, memiliki banyak fitur, dan tidak dipungut biaya (Iswari, 2015). Tujuan dari proyek ini dalam untuk membangun sebuah perangkat lunak pemodelan dan sekaligus platform yang dapat menggantikan perangkat UML berbayar, seperti *Rational Rose, Together*, dan sebagainya.

Dikembangkan dalam bahasa pemrograman Delphi. Walaupun begitu, StarUML merupakan proyek yang *multi – lingual* dan tidak bergantung pada bahasa pemrograman yang spesifik, sehingga bahasa pemrograman apapun dapat digunakan untuk mengembangkan StarUML, seperti C/C++, Java, Visual Basic, Delphi, Jscript, VBScript, C#, VB.NET, dan sebagainya.

### *Unified Modeling Language (UML)*

*Unified Modeling Language* merupakan salah satu metode pemodelan visual yang digunakan dalam perancangan dan pembuatan sebuah *software* yang berorientasikan pada objek. UML adalah sebuah bahasa pemodelan perangkat lunak yang telah distandarisi sebagai media penulisan cetak biru (*blueprints*) perangkat lunak (*pressman*) (Sumiati et al., 2021). UML merupakan sebuah standar penulisan atau semacam *blue print* dimana didalamnya termasuk sebuah bisnis proses, penulisan kelas - kelas dalam sebuah bahasa yang spesifik. Terdapat beberapa diagram UML yang sering digunakan dalam pengembangan sebuah sistem, yaitu:

- a. ***Use Case***: Merupakan gambaran dari fungsionalitas yang diharapkan dari sebuah sistem, dan mempresentasikan sebuah interaksi antara aktor dan sistem. Di Dalam *Use Case* terdapat aktor yang merupakan sebuah gambaran entitas dari manusia atau sebuah sistem yang melakukan pekerjaan di sistem.
- b. ***Activity Diagram***: Merupakan gambaran alir dari aktivitas - aktivitas di dalam sistem yang berjalan.
- c. ***Sequence Diagram***: Menggambarkan interaksi antar objek di dalam dan di

sekitar sistem yang berupa *message* yang digambarkan terhadap waktu.

- d. *Class Diagram*: Merupakan gambaran struktur dan deskripsi dari *class*, *package*, dan objek yang saling berhubungan seperti diantaranya pewarisan, asosiasi dan lainnya.

### Use Case Diagram

*Use case diagram* yaitu model hasil analisis perancangan sistem yang bertujuan untuk mendeskripsikan kebutuhan sistem. Kebutuhan sistem tersebut akan diterapkan oleh pengguna sehingga perancangan sistem dapat tergambarkan (Ocha Widya Susanti & Ummami, 2022). *Use case diagram* adalah abstraksi dari interaksi antara sistem dengan aktor. Oleh karena itu sangat penting untuk memilih abstraksi yang sesuai dengan kebutuhan. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara admin dan *user* dari sebuah sistem dengan melalui sebuah diagram bagaimana sebuah sistem dipakai (Ibnu Alvayet & Vezrino Barrichelo, 2023).

Sebuah *use case diagram* menyatakan visualisasi interaksi yang terjadi antara pengguna (aktor) dengan sistem. Diagram ini bisa menjadi gambaran yang bagus untuk menjelaskan konteks dari sebuah sistem sehingga terlihat jelas batasan dari sistem (Larman dalam Kurniawan, 2018). Ada 2 elemen penting yang harus digambarkan, yaitu aktor dan *use case*. Aktor adalah segala sesuatu yang berinteraksi langsung dengan sistem, bisa merupakan orang (yang ditunjukkan dengan perannya dan bukan namanya/personalnya) atau sistem komputer yang lain. Aktor dinotasikan dengan simbol gambar orang-orangan (*stick-man*) dengan nama kata benda di bagian bawah yang menyatakan peran atau sistem. Aktor bisa bersifat primer, yaitu yang menginisiasi berjalannya sebuah *use case*, atau sekunder, yaitu yang membantu berjalannya sebuah *use case*. *Use case* dinotasikan dengan simbol elips dengan nama kata kerja aktif di bagian dalam yang menyatakan aktivitas dari perspektif aktor. Setiap aktor dimungkinkan untuk berinteraksi dengan sistem dalam banyak *use case*. Sebaliknya, setiap *use case* bisa dijalankan oleh lebih dari satu aktor.

Antar aktor maupun antar *use case* bisa memiliki relasi, masing - masing dengan

spesifikasi yang berbeda. Sebuah *use case*, disebut dengan *base use case*, bisa memiliki relasi dengan satu atau lebih *use case* yang lain, disebut dengan *supplier use case*, dalam bentuk *extend* dan *include*. Relasi *extend* menyatakan bahwa fungsionalitas dari *base use case* bisa diperluas oleh *supplier use case*, jika dibutuhkan, di dalam eksekusi alur alternatif yang ada pada *use case scenario* dari *base use case*. Sedangkan, relasi *include* menyatakan bahwa fungsionalitas dari *base use case* selalu hanya bisa dipenuhi dengan bantuan dari *supplier use case* di dalam eksekusi alur utama yang ada pada *use case scenario* dari *base use case*. Dalam hal ini, relasi *include* dan *extend* tidak menjelaskan urutan eksekusi apapun antara *base use case* dan *supplier use case*, baik dalam alur utama maupun alternatif yang dijelaskan dalam *use case scenario* dari *base use case*. Selanjutnya, sebuah aktor, disebut aktor induk, bisa memiliki relasi dengan aktor yang lain, disebut aktor turunan, yang menyatakan bahwa sebuah aktor merupakan turunan dari aktor yang lain. Aktor turunan akan memiliki hak akses terhadap fungsionalitas sistem yang lebih luas dibandingkan dengan aktor induk.

### Simbol Pada Use Case Diagram

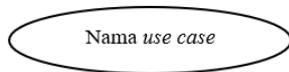
Komponen diagram *use case* yang utama adalah aktor, *use case*, asosiasi dan *stereotype*. *Stereotype* merupakan syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami (Astuti, 2009). Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

- a. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
- b. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit atau aktor.

Berikut adalah simbol - simbol yang ada pada diagram *use case*:

- a. *Use case*: Fungsionalitas yang disediakan sistem sebagai unit – unit yang saling bertukar pesan antar unit

atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*.



Gambar 1. *Use case*

- b. Aktor: Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama aktor. Aktor merupakan peran yang dimainkan oleh pemakai ketika berinteraksi dengan sistem.



Gambar 2. Aktor

- c. Asosiasi atau *association*  
Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.



Gambar 3. Asosiasi atau *association*

- d. Ekstensi atau *extend*  
Relasi *use case* tambahan ke sebuah *use case* dan *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.



Gambar 4. Ekstensi atau *extend*

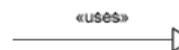
- e. Generalisasi atau *generalization*

Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.



Gambar 5. Generalisasi atau *generalization*

- f. Menggunakan/*include/uses*  
Relasi *use case* tambahan ke sebuah *use case* dan *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini. Ada dua sudut pandang yang cukup besar mengenai *include* di *use case*:
  1. *Include* atau *uses* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan.
  2. *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan.



Gambar 6. *Include* atau *uses*

### Tahapan - tahapan *Use Case Diagram*

Berikut tahapan - tahapan *use case diagram*:

- a. Menemukan aktor  
Pekerjaan awal dalam mendesain sistem adalah menemukan aktor, menemukan fungsionalitas dan membatasi sistem yang akan dibuat. Pembatasan sistem ini penting untuk menemukan aktor. Karena dari sinilah kita akan menemukan apakah sesuatu itu adalah aktor dan apakah aktor tersebut akan berbentuk orang atau sistemlain. Aktor adalah segala hal di luar sistem yang akan menggunakan sistem tersebut untuk melakukan sesuatu.

- b. Menemukan *use case*  
Jika sudah berhasil menemukan aktor, maka untuk menemukan *use case* akan lebih mudah dilakukan. Sebuah *use case* harus mendeskripsikan sebuah pekerjaan dimana pekerjaan tersebut akan memberikan nilai yang bermanfaat bagi aktor.
- c. Membuat *scenario* per *use case*.

**Activity Diagram**

UML (*Unified Modelling Language*) adalah bahasa model standar untuk pengembangan cetak biru perangkat lunak. Salah satu diagram UML adalah *activity diagram*. Pada UML, simbol merupakan *stereotype* aktivitas dari suatu *activity diagram*. Pada *activity diagram*, suatu proses mengambil *input* berupa sumber daya dari sebelah kiri kemudian mengindikasikan *outputnya* pada sebelah kanan.

*Activity diagram* menggambarkan aliran fungsionalisme dalam suatu sistem informasi. Secara lengkap, *activity diagram* mendefinisikan dimana *workflow* dimulai, dimana berakhirnya, aktivitas apa yang terjadi selama *workflow*, dan bagaimana urutan kejadian aktivitas tersebut (Dewi et al., 2017). *Activity diagram* juga menyediakan pendekatan untuk proses pemodelan paralel. Bagi mereka yang akrab dengan analisis dan desain struktur tradisional, diagram ini menggabungkan ide-ide yang mendasari diagram alir data dan diagram alur sistem.

Berikut notasi – notasi pada *Activity Diagram* yang terangkum dalam *Toolbox Enterprise Architect*:



Gambar 7. *Toolbox Enterprise Architect*

Berikut simbol - simbol *Activity Diagram*:

NO	BENTUK SIMBOL	NAMA SIMBOL	FUNGSI SIMBOL
1.		Activity	Menyatakan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
2.		Control Flow	Menunjukkan Urutan Eksekusi.
3.		Object Flow	Menunjukkan aliran objek dari sebuah action atau activity ke action.
4.		Start Point	Menyatakan bahwa sebuah objek dibentuk atau dimulai.
5.		End Point	Menyatakan bahwa sebuah objek dibentuk atau berakhir.
6.		Join/Penggabungan	Menyatakan untuk menggabungkan kembali activity atau action yang paralel.
7.		Fork	Menyatakan untuk memecah behavior menjadi activity atau action yang paralel.
8.		Decision	Menunjukkan penggambaran suatu keputusan/tindakan yang harus di ambil pada kondisi tertentu.

Gambar 8. Simbol - simbol *Activity diagram*

**Class Diagram**

*Class diagram* adalah merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan- aturan dan tanggung jawab entitas yang menentukan perilaku sistem. Jadi dapat dikatakan bahwa *Class Diagram* adalah visual dari struktur sistem program pada jenis-jenis yang di bentuk. *Class Diagram* merupakan alur jalannya sebuah *database* pada *system* yang akan dibangun atau dibuat. *Class diagram* juga disebut kumpulan dari beberapa *class* dan relasinya. *Class* identik dengan *entity* yang direpresentasikan dalam bentuk persegi dimana pada bagian atas ditulis nama *class*, kemudian ke bawah ditulis *attribute* yang terdapat pada *class*, kemudian ke bawah lagi ditulis metode yang ada pada *class*. Sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek.

Selama proses analisis, *class diagram* memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. Selama tahap desain, *class diagram* berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur yang dibuat dapat membantu kita dalam visualisasi struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak dipakai. *Class Diagram* memperlihatkan hubungan antar kelas

dan penjelasan detail tiap- tiap kelas di dalam model desain (dalam *logical view*) dari suatu sistem.

Atribut dan operasi merupakan bagian dari *Class Diagram* yang dapat memberi gambaran hubungan antara perancangan dan perangkat lunaknya sehingga sesuai dengan pembuatan programnya. *Class Diagram* secara khas meliputi: Kelas (*Class*), *Relasi Associations*, *Generalization* dan *Aggregation*, atribut (*Attributes*), operasi (*operation/method*) dan *visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *Multiplicity* atau *Cardinality*. *Class Diagram* menggambarkan serta deskripsi dari *class*, atribut dan objek serta hubungan satu sama lain. *Class Diagram* dapat memberikan pandangan global atas sebuah sistem. Hal tersebut tercermin dari *class* yang ada dan relasinya satu dengan yang lainnya. Sebuah sistem biasanya mempunyai beberapa *class diagram*. *Class Diagram* sangat membantu dalam visualisasi struktur kelas dari suatu sistem. Diagram ini umum digunakan pada pemodelan sistem. *Class Diagram* berfungsi untuk menjelaskan tipe dari objek sistem dan hubungannya dengan objek yang lain.

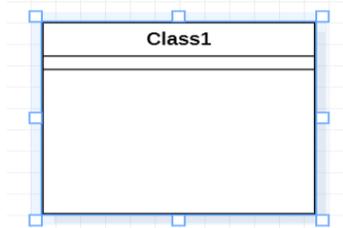
### Simbol Pada *Class Diagram*

*Class Diagram* menggambarkan struktur sistem dari segi pendefinisian kelas - kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

*Class diagram* mempunyai simbol-simbol dalam penggunaannya, yaitu:

#### a. *Class*

*Class* adalah blok - blok pembangunan pada pemrograman berorientasi objek. Sebuah *class* digambarkan sebagai sebuah kotak yang terbagi menjadi 3 bagian.



Gambar 9. *Class*

#### b. *Association*

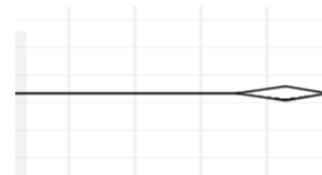
*Association* adalah sebuah hubungan yang menunjukkan adanya interaksi antar *class*. Hubungannya ini dapat ditunjukkan dengan garis dengan mata panah terbuka di ujungnya yang mengindikasikan adanya aliran pesan dalam satu arah.



Gambar 10. *Association*

#### c. *Aggregation*

*Aggregation* mengindikasikan keseluruhan bagian *relationship* dan biasanya disebut relasi.



Gambar 11. *Aggregation*

#### d. *Composition*

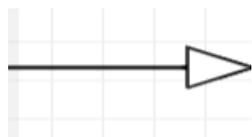
Jika sebuah *class* tidak bisa berdiri sendiri dan harus merupakan bagian dari *class* yang lain, maka *class* tersebut memiliki relasi *Composition* terhadap *class* tempat dia bergantung tersebut. Sebuah *relationship composition* digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/solid.



Gambar 12. *Composition*

#### e. *Generalization*

*Generalization* adalah sebuah hubungan antar *class* yang bersifat dari khusus ke umum.



Gambar 13. *Generalization*

- f. *Dependency*  
Kadangkala sebuah *class* menggunakan *class* yang lain. Hal ini disebut *dependency*. Umumnya penggunaan *dependency* digunakan untuk menunjukkan operasi pada suatu *class* yang menggunakan *class* yang lain. Sebuah *dependency* dilambangkan sebagai sebuah panah bertitik-titik.



Gambar 14. *Dependency*

### Komponen *Class Diagram*

*Class Diagram* dibagi menjadi tiga komponen, yaitu komponen tengah, komponen atas, dan komponen bawah. Komponen tengah adalah tempat semua operasi didefinisikan dan sedangkan komponen atas dan bawah hanya berisi atribut.

Berikut ini adalah penjelasan terkait ketiga komponen tersebut.

- a. **Komponen Tengah**  
Komponen tengah dalam *class diagram* merupakan komponen yang paling penting, karena menggambarkan sebuah karakteristik objek. Ini termasuk data anggota dan juga berisi metode yang menjelaskan bagaimana suatu objek berinteraksi dengan objek atau proses lain.
- b. **Komponen Atas**  
Komponen atas dalam komponen *Class Diagram* adalah *class*. Ini mewakili komponen tingkat tertinggi dari sistem. Suatu *class* dapat berisi *class* lain sebagai anggotanya dan juga dapat menjadi anggota *class* lain.
- c. **Komponen Bawah**  
Komponen ini mewakili kumpulan objek yang berbagi atribut dan perilaku tertentu. Di UML, *class* paling bawah dalam diagram disebut *class root*. *Class root* ini terlihat seperti lingkaran dengan garis yang melewatinya secara vertikal

### METODE PENELITIAN

#### Identifikasi Masalah

Tahapan Pertama pada penelitian ini merupakan identifikasi masalah. Identifikasi masalah ini adalah tahapan dimana kami menemukan suatu permasalahan yang muncul pada perpustakaan SMPN 13 Tambun Selatan. Dengan tahapan ini, kami mengidentifikasi untuk membuat sistem informasi perpustakaan berbasis *website*.

#### Pengumpulan Data

Selanjutnya yaitu tahapan pengumpulan data yang dimana tahapan ini memiliki 3 tahapan yang berbeda yaitu studi literatur, wawancara, dan observasi. Studi literatur adalah metode pengumpulan informasi/data untuk mencari referensi pendukung penelitian dengan menelaah berbagai buku yang berkaitan dengan hal yang diteliti. Observasi adalah proses yang menjelaskan kemampuan mengamati dan mengingat. Pada penelitian ini dilakukan secara langsung pada perpustakaan SMPN 13 Tambun Selatan. Pada tahapan wawancara, kami melakukan kegiatan tanya jawab secara langsung terhadap Rasyid sebagai pengelola perpustakaan di SMPN 13 Tambun Selatan.

#### Analisis Sistem

Tahapan ini dilakukan untuk menganalisis efektivitas dan efisiensi pembuatan sistem informasi perpustakaan berbasis *website*. Analisis sistem ini dilakukan berdasarkan hasil observasi serta wawancara yang telah dilakukan. Dari hasil tersebut, kami memberikan gambaran yang sangat jelas untuk melakukan perancangan sistem informasi perpustakaan berbasis *website*.

#### Desain Sistem

Tahapan Desain sistem bertujuan untuk memberikan gambaran yang jelas dan lengkap yang sebagian besar berorientasi pada komputer. Pada tahapan ini, sistem akan dirancang menggunakan *class diagram*.

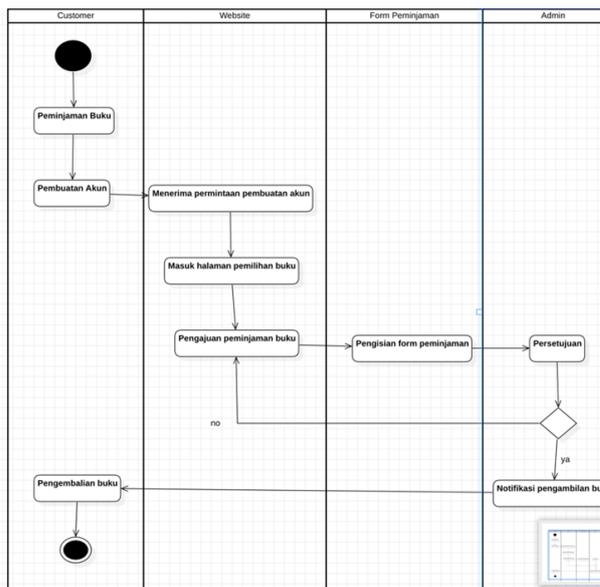
### HASIL DAN PEMBAHASAN

Melalui perancangan sistem ini, kami merancang sebuah sistem untuk pengelolaan perpustakaan SMPN 13 Tambun selatan dengan

menggunakan StarUML lalu akan dikembangkan dan disempurnakan menggunakan *software* lain. Dengan demikian akan tercipta sistem informasi perpustakaan yang efektif dan efisien. Berikut adalah tahapan merancang sebuah sistem informasi:

### Activity Diagram

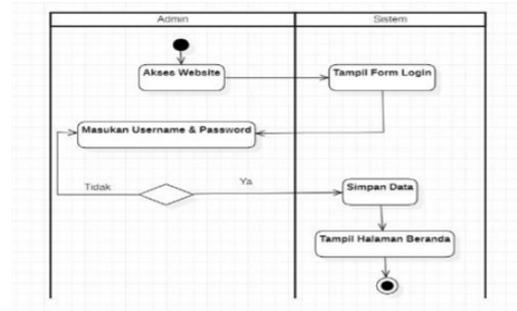
Berikut adalah hasil *Activity diagram* untuk sistem peminjaman buku secara *online*:



Gambar 15. *Activity diagram* sistem peminjaman buku secara online

Gambar di atas menunjukkan proses peminjaman buku di perpustakaan *online*, pertama pada kolom pelanggan tertulis mulai dan ingin memulai peminjaman buku di perpustakaan *online*, kemudian pada kolom daftar atau buat akun, kunjungi *website* yang tersedia untuk peminjaman dan pilih buku yang tersedia, jika anda menemukan buku dan ingin meminta pinjaman, maka masuk ke kolom formulir yang berisi informasi tentang nama buku dan jangka waktu pinjaman, anda akan masuk ke kolom tata usaha yang berisi izin peminjaman perpustakaan. Jika iya maka admin akan mengirimkan notifikasi penerimaan buku, jika tidak maka nasabah akan dialihkan ke kolom permintaan pinjaman untuk menyelesaikan proses, jika iya maka nasabah akan mengambil buku tersebut dari perpustakaan dan itu saja.

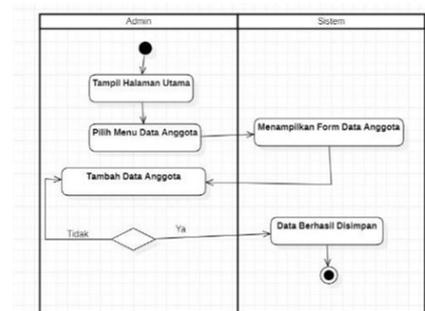
### Activity Diagram Login Admin



Gambar 16. *Activity diagram* login admin

Admin mengakses *website* kemudian admin akan melakukan *login* terlebih dahulu dengan memasukkan *username* dan *password*, selanjutnya apabila *username* dan *password* sudah sesuai, sistem dapat menampilkan halaman beranda. Selain itu, apabila *username* dan *password* tidak sesuai, maka sistem dapat mengarahkan untuk mengisi kembali *username* dan *password*.

### Activity Diagram Anggota

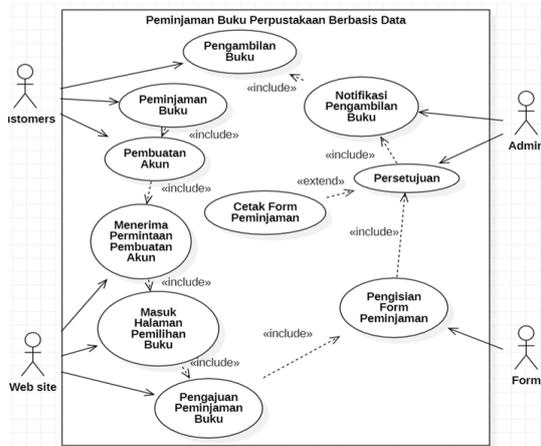


Gambar 17. *Activity diagram* anggota

Pada *activity diagram* data anggota. terdapat Sistem menampilkan halaman utama, lalu admin pilih menu data anggota kemudian sistem dapat menampilkan *form* data anggota, lalu admin akan mengisi *form* tambah data anggota kemudian mengklik tombol *submit*. Kemudian sistem dapat melakukan validasi, jika berhasil sistem dapat menampilkan sebuah pesan data anggota berhasil disimpan, jika ada data yang belum diisi maka sistem dapat menampilkan *form* untuk admin dapat mengisi kembali data anggota dengan benar.

### Use Case Diagram

Berikut adalah hasil *Use Case diagram* untuk sistem peminjaman buku secara *online*:



Gambar 18. *Use case diagram* sistem peminjaman buku secara online

Pada gambar menunjukkan *use case diagram* yang dimana sebuah sistem informasi peminjaman buku perpustakaan berbasis *online* yang dimana *customer* akan meminjam buku perpustakaan. Berikut adalah langkah-langkah yang dilakukan:

1. *Customer*: merupakan aktor yang bertujuan untuk peminjaman buku, pembuatan akun *website*, dan melakukan pengambilan buku di perpustakaan.
2. *Website*: merupakan aktor yang bertujuan untuk menerima pembuatan akun, masuk pada halaman pemilihan buku, setelah memilih buku lalu melakukan pengajuan peminjaman buku.
3. *Form*: merupakan aktor yang bertujuan untuk pengisian data atau *form* peminjaman.
4. *Admin*: merupakan aktor yang bertujuan untuk menerima persetujuan peminjaman buku, apabila setuju yang nantinya mencetak form peminjaman dan memberikan notifikasi pengambilan buku kepada *customer* yang nantinya *customer* sendiri menerima

notifikasi dan akan mengambil buku tersebut.

Berikut *use case* pada setiap aktor:

#### 1. Customer

- a. Pengambilan buku: Aktor melakukan pengambilan di perpustakaan.
- b. Peminjaman buku: Aktor melakukan peminjaman buku di perpustakaan.
- c. Pembuatan akun: Aktor membuat akun pada *website*

#### 2. Website

- a. Menerima pembuatan akun: *website* menerima pembuatan akun *customer*.
- b. Masuk halaman pemilihan buku: *website* memasuki halaman untuk pemilihan buku.
- c. Pengajuan buku: aktor melakukan pengajuan buku.

#### 3. Form Pengisian

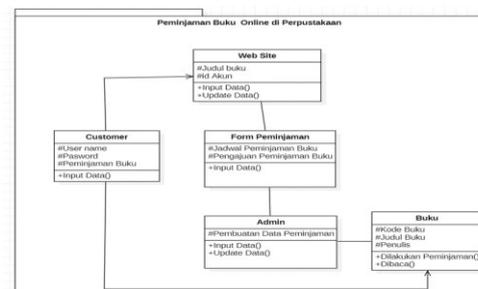
- a. *form* pengajuan: aktor melakukan pengisian data untuk pengajuan pengambilan buku.

#### 4. Admin

- a. Persetujuan: aktor melakukan persetujuan untuk peminjaman buku
- b. Notifikasi pengambilan buku: aktor memberikan notifikasi pengambilan buku pada *customer*.

### Class Diagram

Berikut adalah hasil *Class diagram* untuk sistem peminjaman buku secara *online*:



Gambar 19. *Class diagram* sistem peminjaman buku secara online

Pada gambar di atas menunjukkan *class diagram* yang dimana sebuah sistem informasi peminjaman buku perpustakaan berbasis *online* yang dimana *customer* akan meminjam buku perpustakaan. Berikut adalah pembahasan pada setiap kelas yang dikelola:

1. *Customer*
  - a. *Menginput Username*
  - b. *Menginput Password*
  - c. *Menginput Peminjaman Buku*
2. *Web Site*
  - a. *Menginput Id akun*
  - b. *Mengupdate Judul Buku*
3. *Form Peminjaman*
  - a. *Menginput Peminjaman Buku*
  - b. *Menginput Pengajuan Peminjaman Buku*
4. *Admin*
  - a. *Menginput dan mengupdate pengajuan peminjaman buku*
5. *Buku*
  - a. *Membaca dan dilakukannya peminjaman kode buku untuk customer*
  - b. *Membaca dan dilakukannya peminjaman judul buku untuk customer*
  - c. *Membaca penulis buku untuk customer.*

## KESIMPULAN DAN SARAN

### KESIMPULAN

Berdasarkan hasil penelitian mengenai sistem informasi peminjaman buku perpustakaan berbasis *online*, beberapa kesimpulan dapat diambil. Pertama, mahasiswa telah memahami konsep dan komponen-komponen dalam *Class Diagram*. Kedua, mahasiswa mampu membaca dan menginterpretasikan *Class Diagram* dari suatu sistem informasi dengan baik. Ketiga, mahasiswa memiliki keterampilan untuk membuat *Class Diagram* yang berguna dalam merancang sistem informasi. Hasil penelitian ini menunjukkan bahwa mahasiswa tidak hanya menguasai teori, tetapi juga dapat menerapkan keterampilan praktis dalam perancangan sistem informasi.

## SARAN

Berdasarkan hasil penelitian yang telah dilakukan mengenai sistem informasi peminjaman buku perpustakaan berbasis *online*, beberapa saran telah diperoleh. Pertama, karena pentingnya pengelolaan data perpustakaan berbasis *online* yang baik, dibutuhkan sumber daya manusia yang kompeten selain sistem informasi yang akurat. Hal ini bertujuan untuk mempermudah akses dalam proses peminjaman buku secara *online* di perpustakaan dan mendukung kelancaran operasional sistem perpustakaan. Kedua, disarankan agar penelitian selanjutnya dapat membuat simulasi dari sistem informasi peminjaman buku *online* pada perpustakaan yang telah diteliti.

## DAFTAR PUSTAKA

- Astuti, R. (2009). Pemodelan Analisis Berorientasi Objek dengan Use Case. *Media Informatika*, 8(2), 73–81. [https://jurnal.likmi.ac.id/Jurnal/7\\_2009/Pemodelan\\_Analisis\\_rini\\_.pdf](https://jurnal.likmi.ac.id/Jurnal/7_2009/Pemodelan_Analisis_rini_.pdf)
- Dewi, L. P., Indahyanti, U., & S, Y. H. (2017). Pemodelan Proses Bisnis Menggunakan Activity Diagram Uml Dan Bpmn ( Studi Kasus Frs Online ). *Informatika*, 1–9.
- Fitriyana, F., & Sucipto, A. (2020). Sistem Informasi Penjualan Oleh Sales Marketing Pada Pt Erlangga Mahameru. *Jurnal Teknologi Dan Sistem Informasi*, 1(1), 105–110. <https://doi.org/10.33365/jtsi.v1i1.239>
- Gede Endra Bratha, W. (2022). Literature Review Komponen Sistem Informasi Manajemen: Software, Database Dan Brainware. *Jurnal Ekonomi Manajemen Sistem Informasi*, 3(3), 344–360. <https://doi.org/10.31933/jemsi.v3i3.824>
- Ibnu Alvayet, T. A., & Vezrino Barrichelo, E. (2023). Perancangan Sistem Informasi Pengolahan Data Laporan Pajak Bulanan Berbasis Web Pada Depo Unilever Padang. *Jurnal Sains Informatika Terapan*, 2(3), 108–113. <https://doi.org/10.62357/jsit.v2i3.202>
- Iswari, N. M. S. (2015). Review Perangkat Lunak StarUML Berdasarkan Faktor Kualitas McCall. *Ultimatics : Jurnal*

- Teknik Informatika*, 7(1).  
<https://doi.org/10.31937/ti.v7i1.352>
- Kurniawan, T. A. (2018). Pemodelan Use Case (UML): Evaluasi Terhadap beberapa Kesalahan dalam Praktik. *Jurnal Teknologi Informasi Dan Ilmu Komputer*, 5(1), 77–86.  
<https://doi.org/10.25126/jtiik.201851610>
- Ocha Widya Susanti, E., & Ummami, I. (2022). Rancang Bangun Sistem Informasi Jurnal Perkuliahan Berbasis Web Guna Meningkatkan Efektivitas Pembelajaran. *Jurnal Teknologi Dan Sistem Informasi Bisnis-JTEKSIS*, 4(1), 386.  
<https://doi.org/10.47233/jteksis.v4i2.556>
- Sandfreni, S., Ulum, M. B., & Azizah, A. H. (2021). Analisis Perancangan Sistem Informasi Pusat Studi Pada Fakultas Ilmu Komputer Universitas Esa Unggul. *Sebatik*, 25(2), 345–356.  
<https://doi.org/10.46984/sebatik.v25i2.1587>
- Sumiati, M., Abdillah, R., & Cahyo, A. (2021). *Pemodelan UML untuk Sistem Informasi Persewaan Alat Pesta*. 11(2), 79–86.