BAB II

TINJAUAN PUSTAKA

2.1. Landasan Teori

2.1.1 Pengertian Sistem

Secara umum, sistem dapat diartikan sebagai sebuah himpunan prosedur yang saling terhubung dan terintegrasi, bekerja secara kolaboratif dan terstruktur untuk memperoleh tujuan yang telah ditentukan (Anggraini et al., 2020). Di samping itu, sistem juga dapat diartikan sebagai kumpulan dari beberapa elemen yang bekerja secara sinergis dan saling terkait satu sama lain untuk memproses input berupa data, dengan tujuan mencapai suatu hasil atau tujuan tertentu (Ayu & Permatasari, 2018). Serupa dengan definisi sebelumnya, sistem diartikan sebagai kumpulan objek yang saling berinteraksi secara fungsional dan memiliki relasi yang menghubungkan karakteristik dari setiap elemen objek, agar dapat mencapai tujuan yang diharapkan (Wantoro et al., 2021).

Berdasarkan beberapa pengertian tersebut, maka disimpulkan bahwa sistem adalah serangkaian proses yang terdiri atas prosedur, elemen, dan objek yang saling terhubung agar dapat mencapai tujuan yang diharapkan .

Dalam sebuah buku karangan (Sutabri, 2012) yang berjudul "*Analisis Sistem Informasi"* mengatakan bahwa Sistem memiliki karakteristik atau sifat-sifat yang khas yang menandakan bahwa suatu entitas dapat dianggap sebagai sistem. Berikut adalah karakteristik sistem yang dimaksud:

a. Komponen Sistem (*Components*)

Suatu sistem terdiri dari beberapa bagian yang saling berinteraksi, bekerja secara kolektif bergabung dan terdiri dari subsistem-

subsystem yang memiliki tugas khusus dalam menjalankan tungas tertentu yang berpengaruh kepada seluruh proses sistem.

b. Batasan Sistem (*Boundary*)

   Batasan sistem adalah area yang memisahkan satu sistem dari sistem lain atau lingkungan eksternalnya, dan mencerminkan bahwa suatu sistem merupakan entitas yang terpadu dan tidak dapat dipisahkan.

c. Lingkungan Luar Sistem (*Environtment*)

   Lingkungan ini merujuk pada segala jenis yang berada di luar wilayah atau batasan sistem yang memiliki pengaruh terhadap operasional sistem tersebut. Lingkungan luar dapat memberikan manfaat dan mendukung operasional sistem, namun juga mungkin memberikan dampak negatif yang perlu dikendalikan.

d. Penghubungan Sistem (*Interface*)

   Interface adalah media yang digunakan untuk menyambungkan antara sistem dan subsistem, sehingga aliran sumber daya antar subsistem dapat terjadi dan menciptakan integrasi yang baik di antara mereka.

e. Masukan Sistem (*Input*)

   Input atau masukan dapat dianggap sebagai energi yang dibutuhkan ileh sistem. beberapa jenis masukan sistem masukan masukan sinyal (*signal input*) dan pemeliharaan (*maintenance input*). Sebagai contoh, program yang dijalankan di komputer merupakan contoh masukan pemeliharaan, sedangkan data yang diolah menjadi informasi merupakan contoh masukan sinyal.

f. Keluaran Sistem (*Output*)

   setelah selesai mengolah *input* maka akan menghasilkan *output* yang dapat menjadi masukan bagi subsistem lain dalam sistem yang sama. Dapat dikatakan bahwa hasil keluaran dari suatu sistem dapat menjadi masukan untuk subsistem lain dalam sistem yang lebih besar.

g. Pengolahan Sistem (*Process*)

Untuk menghasilkan suatu hasil atau keluaran sistem yang diolah dari masukan *input*, maka suatu sistem membutuhkan suatu proses sistem.

h. Sasaran Sistem (*Objective*)

Sasaran atau tujuan dari sebuah sistem tentunya telah ditentukan sebelumnya, ketika sasaran atau tujuan ini dapat terpenuhi dengan baik oleh sistem, maka sistem tersebut dapat dikatakan barhasil.

2.1.2 Pengertian Informasi

Data yang telah diolah sehingga menghasilkan informasi yang memiliki nilai dan makna yang lebih berguna bagi pengguna. Informasi memberikan pemahaman tentang suatu kejadian atau situasi yang berguna dalam pengambilan keputusan (Rusdiana, 2018). Informasi juga dapat dikonsepsikan sebagai proses transformasi data menjadi pengetahuan yang dapat dipahami oleh manusia melalui panca inderanya. Selain itu, informasi juga dapat saling dipertukarkan antara individu atau sistem untuk keperluan komunikasi dan pertukaran pengetahuan (Sri et al., 2018).

2.1.3 Pengertian Sistem Informasi

Sistem yang digunakan oleh sebuah organisasi ataupun pihak luar guna mendapatkan sebuah laporan yang bersifat manajerial yang didapatkan dari pengolahan transaksi harian, serta dapat mendukung operasional suatu organisasi adalah pengertian Sistem informasi menurut (Dewi et al., 2021). Sedangkan menurut (Yanuarsyah & Napianto, 2021) Sistem Informasi adalah gabungan yang melibatkan manusia (*human*), jaringan komunikasi (*network*), perangkat keras (*hardware*), perangkat lunak (*software*), dan data menjadi sumber daya utama. Sistem ini dilengkapi dengan prosedur untuk membuat (*create*), membuka atau mengakses (*read*), mengubah (*update*), dan menghapus (*delete*) data. Informasi yang dihasilkan oleh sistem tersebut dapat dimanfaatkan oleh organisasi.

2.1.4 Pengertian System Development Lifecycle

System Development Lifecycle (SDLC) atau dalam bahasa Indonesia berarti siklus hidup pengembangan sistem adalah suatu proses yang digunakan untuk membuat atau mengembangkan *software* dengan model dan metodologi yang telah terbukti efektif dan banyak digunakan (Putra & Andriani, 2019). Dalam sebuah jurnal karangan (Julianto & Setiawan, 2019) mengatakan bahwa SDLC adalah sebuah kerangka kerja atau *framework* yang telah disusun sedemikian rupa sehingga memilki struktur yang berurutan dan digunakan untuk pengembangan sistem informasi.

Berdasarkan pengertian tersebut, maka SDLC adalah suatu kerangka kerja yang melibatkan langkah-langkah dan rencana yang komprehensif untuk membangun, merawat, dan mengganti suatu perangkat lunak. SDLC harus dijalankan untuk memproses pengembangan sistem perangkat lunak.

Tujuan penggunaan SDLC adalah memastikan bahwa sistem informasi yang dibangun akan sesuai dengan kebutuhan dan harapan organisasi. Melalui penerapan SDLC, diharapkan masalah dalam pembuatan atau pengembangan sistem informasi dapat diselesaikan dengan efektif, dan menghasilkan sistem yang berkualitas tinggi. Tahapan dalam SDLC adalah sebagai berikut:

1. Perancangan (*Planning*)

   Dalam tahap ini, fokus utamanya adalah pada studi kelayakan (feasibility study) dalam pengembangan sistem. Tim pengembang akan dibentuk, tujuan dan ruang lingkup pengembangan akan ditetapkan, strategi akan ditentukan dan dievaluasi, prioritas teknologi akan ditentukan, serta alokasi sumber daya, perencanaan kapasitas, jadwal proyek, dan estimasi biaya akan dilakukan. Tahap ini akan menghasilkan rencana proyek, jadwal pelaksanaan, estimasi biaya, dan aturan yang berlaku selama proyek pengembangan sistem berlangsung.

2. Analisis (*Anlysis*)

   Tahap analisis adalah saat di mana sistem yang akan dibangun akan dianalisis tentang bagaimana cara kerjanya dan juga menentukan spesifikasinya. Tim pengembang akan melakukan studi literatur tentang kasus yang akan diatasi oleh sistem, melakukan brainstorming tentang masalah yang akan dimodelkan oleh sistem, mengelompokan masalah, kesematan, dan penyelesaian yang akan ditawarkan untuk masalah tersebut.

   Selain itu, keinginan terhadap sistem juga perlu ditentukan, dianalisis, dan diuraikan agar tim pengembang dapat menetapkan batasan sistem yang akan dibangun. Setelah tahap analisis ini, diharapkan kelebihan dan kekurangan sistem, fungsi-fungsi yang akan diimplementasikan, serta kebutuhan pemeliharaan sistem dapat diidentifikasi.

3. Desain (*Design*)

   Berdasarkan hasil yang didapatkan dari tahap analisis sebelumnya, maka akan diketahui *blueprint* atau rancangan sistem yang siap dikembangkan, karakteristik dan langkah tindakan yang akan ada pada sistem nantinya akan dijelaskan secara rinci sehingga memudahkan tim pengembang untuk membangun sistem tersebut. Dalam tahap ini tim akan menganalisis hubungan antara objek dan fungsi pada sistem, data juga akan dianalis pada tahap ini untuk kemudian dibuatkan skema basis data nya, setelah semua terdefinisakan maka selanjutnya akan merancang antar muka pengguna untuk sistem. Pada tahap desain, akan dihasilkan purwarupa dan beberapa hasil lainnya berupa dokumen tentang desain sistem, desain implementasi, dan daftar komponen yang diperlukan.

4. Implementasi (*Implementation*)

   Pada tahap implementasi, pengembang akan akan mengimplementasikan *prototype* dari sistem yang dihasilkan dari tahap desain untuk melakukan uji coba penggunaan sistem, untuk

memulai melakukan implementasi sistem, maka *database* yang dibutuhkan perlu dibuat dengan memperhatikan skema yang telah dibuat sebelumnya, kemudian aplikasi akan mulai dibuat dengan desain sistem yang telah dibuat menjadi dasar pembuatannya kemudian sistem yang telah dibuat akan dilakukan pengujian dan perbaikan aplikasi (*debugging)* atas masalah yang mungkin ditemukan. Setelah semua masalah yang ditemukan telah diperbaiki, maka perangkat lunak telah sesuai dengan kebutuhan dan spesifikasi telah siap untuk dilakukan pengujian hingga integrasi.

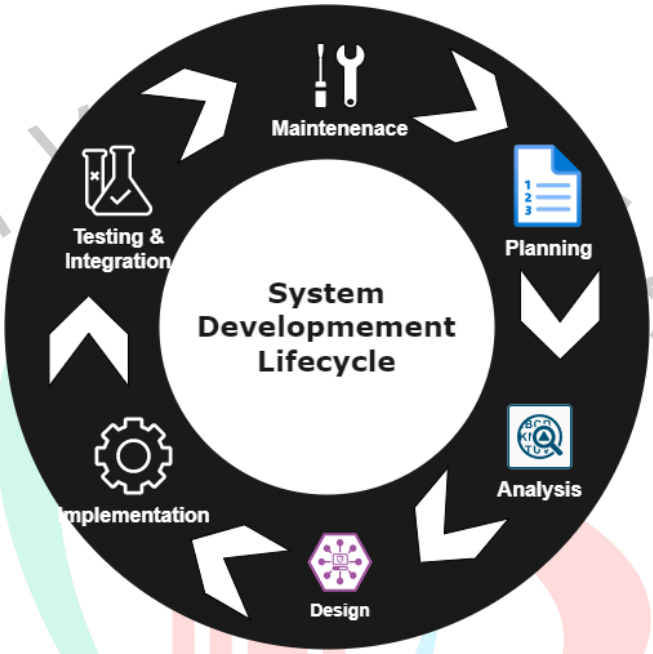
5. Pengujian dan integrasi (*Testing and integration*)

   Tahap pengujian dan integrasi adalah tahapan yang paling penting dalam proses pembuatan sebuah program, sebuah program atau sistem yang telah dibangun haruslah dilakukan pengujian sebelum sistem dipublikasikan. Kualitas dari penulisan kode dan kemudahan penggunaan sistem dari segi fungsionalistas, integrasi, performa dan keamanan akan diuji dengan teratur dan teliti sehingga tidak ada bagian dari sistem yang terlewat. Setelah selesai diuji maka sistem atau program akan siap untuk dipublikasikan dalam proses produksi.

6. Pemeliharaan (*Maintenance*)

   Setelah semua tahap sebelumnya selesai makan tahapan akhir dari pengembangan sebuah sistem adalah tahap pemeliharaan atau *maintenance*, sistem akan dipantau secara berkelanjutan untuk memastikan sistem bekerja sebagai mana mestinya sesuai desain awal. Semua aktivitas pemeliharaan perlu dicatat untuk menjadi rujukan di kemudian hari, jika saat pemeliharaan ditemukan celah ataupun kerusakan pada sistem akan segera dilaporkan untuk kemudian dilakukan perbaikan pada semua celah atau kerusakan.

Secara umum, tahapan SDLC adalah perencanaan, pengamatan, perancangan, pelaksanaan, pengetesan dan integrasi, serta perawatan. Berikut adalah **Gambar 2.1** mengenai Informasi lebih detail tentang tahapan tersebut:



Gambar 2.1 Tahapan System Development Life Cycle

2.1.5 Pengertian Unified Modeling Language

Unified Modeling Language (UML) adalah suatu standar bahasa yang dimanfaatkan dalam proses pemodelan aplikasi yang akan dikembangkan (Putra & Andriani, 2019). UML digunakan untuk menggambarkan kebutuhan, uraian, desain sistem, serta mendefinisikan konstruksi dalam pemrograman yang berdasarkan objek (Yanuarsyah & Napianto, 2021), UML dapat diartikan sebagai bahasa standar yang dimanfaatkan untuk membuat rancangan atau cetak biru dari sistem atau perangkat lunak. UML memungkinkan penggambaran, penentuan, dan dokumentasi model yang membantu dalam proses pengembangan perangkat lunak.

Sejalan dengan pernyataan sebelumnya, (Ciccozzi et al., 2019) menjabarkan bahwa UML bahasa pemodelan perangkat lunak utama

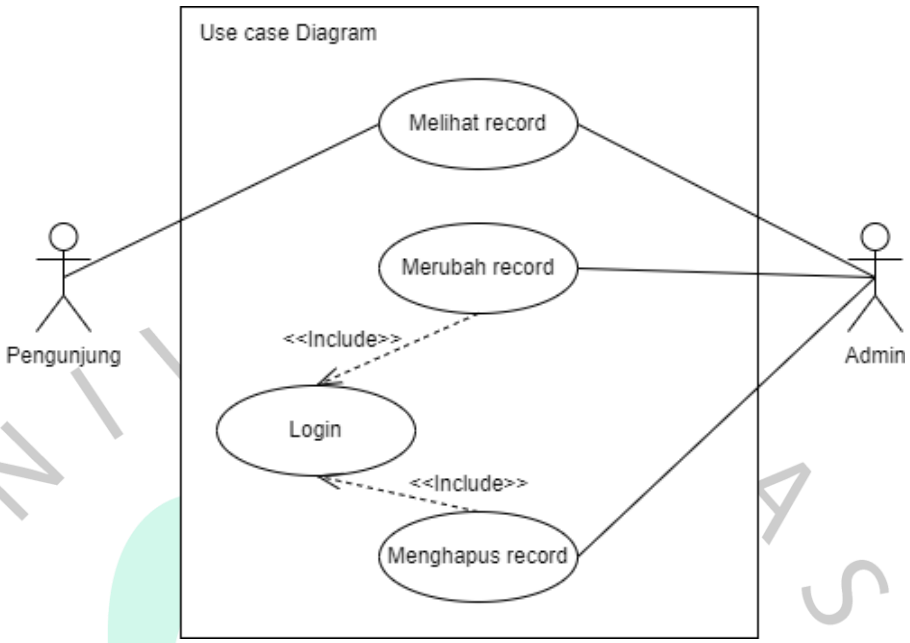
yang menetapkan diri secara *de facto* dan *de jure* karena memiliki fungsi yang serbaguna, UML dapat digunakan sebagai bahasa pemodelan umum ataupun pemodelan khusus dengan menyesuaikan melalui pembuatan profilnya.

Berdasarkan pernyataan terkait UML diatas disimpulkan bahwa UML adalah sebuah metode pemodelan dalam perancangan sebuah perangkat lunak berorientasi objek yang disajikan secara visual. Beberapa diagram yang umumnya digunakan adalah:

2.1.5.1 Use Case Diagram

Use case diagram adalah metode yang digunakan dalam analisis sistem untuk mengidentifikasi, menjelaskan, dan mengatur kebutuhan sistem. diagram ini merupakan diagram dalam UML dan termasuk dalam kategori diagram perilaku (*behavioral diagram*). diagram ini digunakan untuk memberikan gambaran bagaimana sistem berinteraksi dengan entitas internal atau eksternal (Aleryani, 2016). Selain itu, diagram ini juga merupakan salah satu bentuk diagram yang dapat memvisualisasikan hubungan antara aktor dengan sistem. Setiap *use case* dalam diagram tersebut menggambarkan spesifikasi aktivitas atau fungsi sistem yang dibutuhkan oleh aktor untuk memenuhi kebutuhannya terhadap sistem, juga mengilustrasikan kebutuhan sistem yang ada (Kurniawan, 2018).

Use case diagram merupakan sebuah diagram yang memvisualisasikan fungsi pengguna dalam menggunakan sistem tertentu, dan berfungsi sebagai alat visualisasi untuk menggambarkan hubungan antara pengguna dan sistem tersebut. Diagram ini juga memberikan gambaran tentang spesifikasi kasus penggunaan fungsi dalam sistem (Destriana et al., 2021). Berikut adalah **Gambar 2.2** contoh *use case* diagram sederhana.

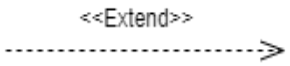
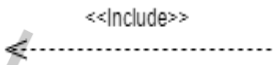


Gambar 2.2 Contoh *use case diagram*

Use case diagram tersusun atas beberapa komponen yang berfungsi untuk menggambarkan hubungan antara pengguna dengan sistem, di antaranya adalah:

Tabel 2.1 *Komponen Usecase Diagram*

Deskripsi	Simbol
Use case, menggambarkan fungsi-fungsi dalam sistem sebagai elemen yang saling berhubungan atau dengan aktor-aktor yang terlibat.	Use Case
Actor, menggambarkan individu, prosedur, atau sistem eksternal lainnya yang akan berhubungan dengan sistem yang sedang dibangun.	Aktor/Role
Association, mengilustrasikan interaksi komunikasi antara use case dan aktor yang terlibat dalam interaksi dalam sistem.	

Extend, menunjukkan keterkaitan antara use case utama dengan use case pelengkap, di mana use case pelengkap dapat diakses secara independen tanpa harus melalui use case utama.	
Include, adalah keterkaitan di antara use case utama dengan use case pelengkat yang menjadi prasyarat agar use case utama bisa dijalankan.	
Generalization, keterkaitan umum dan khusus antara aktor atau use case mengacu pada situasi di mana ada aktor atau use case yang memiliki fungsi yang sama dengan yang lainnya, tetapi juga memiliki fungsi tambahan yang hanya dimiliki oleh aktor atau use case tertentu.	
Boundary, menampilkan batasan antara sistem dengan aktor diluar sistem.	

Sumber: Pemodelan Analisis Berorientasi Objek Dengan Use Case,(Astuti, 2009)

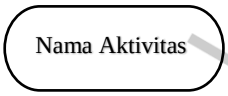
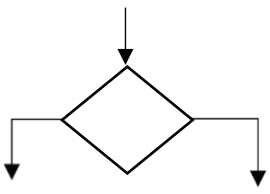
2.1.5.2 Activity Diagram

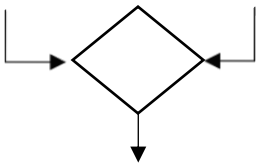
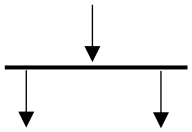
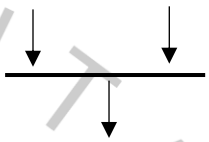
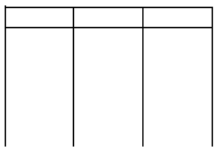
Activity diagram adalah sebuah diagram yang memvisualisasikan urutan kerja dalam suatu bisnis dengan menggunakan alur dasar dan mungkin memiliki alur kerja alternatif. Alur kerja dapat didefinisikan secara teks dengan menggunakan pernyataan *if, if-then-else*, atau *do-until*. Diagram ini berguna dalam pemodelan sistem di berbagai domain (Ericsson, 2003). Sedangkan pengertian *activity diagram* menurut (Putra & Andriani, 2019) adalah diagram yang dapat memvisualisasikan *work flow* atau aktivitas dari sistem yang ada pada sebuah program atau *softeware*.

Sejalan dengan pengertian diatas, menurut (Aliman, 2021) *activity diagram* Merupakan model yang menggambarkan bagaimana objek atau sistem bekerja, di mana diagram ini menampilkan alur kerja terstruktur dari proses kerja use case mulai dari awal hingga akhir. Setiap aktivitas dalam diagram ini digambarkan menggunakan simbol-simbol khusus yang menjelaskan fungsinya. Dari beberapa penjelasan di atas, maka *activity diagram* adalah sebuah diagram yang memvisualisasikan proses yang terjadi dalam sebuah sistem. Diagram ini memvisualisasikan bagaimana aktivitas dimulai, keputusan dan proses paralel yang kemungkinan terjadi, serta bagaimana aktivitas tersebut selesai.

Activity diagram memliki simbol dan notasi yang digunakan untuk menggambarkan aktivitas, *object*, *decision*, dan lain-lain. Simbol tersebut dijelaskan dalam **Tabel 2.2** dibawah ini.

Tabel 2. 2 Simbol Activity Diagram

Keterangan	Simbol
Start state, Menandakan awal dimulainya alur kerja dalam activity diagram dan hanya ada satu titik awal dalam satu alur kerja.	
Action/Activity, menggambarkan sebuah aktivitas atau tugas dalam sebuah alur kerja dan tidak dapat diuraikan.	
Object, mewakili suatu objek yang dihasilkan oleh sebuah aktivitas yang dilakukan pada proses sebelumnya.	
Action Flow, menunjukan urutan aliran kerja selanjutnya dari sebuah *action* atau objek.	
Decision and brancing, mengindikasikan suatu kondisi yang memungkinkan adanya perbedaan transisi sesuai dengan kondisi yang telah ditentukan.	

Merge node, digunkan untuk menyatukan kembali aliran keputusan yang yang berbeda.	
Fork node, digunakan untuk membagi satu masukan aktivitas dan membaginya menjadi beberapa aktivitas secara paralel	
Join node, berfungsi untuk menggabungkan beberapa aliran aktivitas paralel sehingga menjadi aktivitas tunggal.	
End state, menandakan berahirnya semua aliran kerja dalam suatu aktivitas.	
Swimline, digunakan untuk membagi *activity diagram* untuk membantu mempermudah pembacaan diagram dan dibagi sesuai dengan aktor atau penanguung jawab aktivitas.	

Sumber: Activity Diagram,(Smartdraw, 2023)

2.1.5.3 Class Diagram

Class diagram adalah representasi visual dari bentuk sebuah sistem yang meliputi definisi kelas-kelas yang menyusun sistem tersebut, termasuk atribut-atribut dan operasi-operasi yang dimiliki oleh setiap kelas. Diagram ini memberikan panduan kepada pengembang untuk membangun sistem sesuai dengan desain yang telah ditetapkan (Putra & Andriani, 2019). Definisi lain dari *Class diagram* adalah sebuah diagram yang memvisualisasikan relasi antara objek-objek dalam sistem, sehingga sistem dapat beroperasi sesuai dengan yang diharapkan (Ayu & Permatasari, 2018).

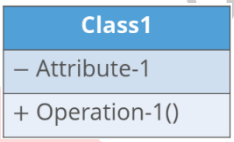
Sedangkan menurut (Ergashev, 2022) mendefinisikan *class diagram* sebagai sebuah diagram yang dapat memetakan struktur sebuah sistem yang mewakili hubungan antara kelas, atribut, operasi, dan objek. Pengembang dapat memodelkan interaksi yang

terjadi dalam sistem sesuai dengan rancangan fungsionalitas kelas, atribut, dan tipe parameter.

Dari beberapa penjelasan diatas, maka disimpulkan bahwa class diagram adalah sebuah diagram dalam UML yang digunakan untuk menggambarkan model statis sistem. Diagram ini memvisualisasikan kelas-kelas dan relasi antar kelas yang ada dalam sistem, yang tetap selama proses hidup sistem.

Komponen yang ada pada *class diagram* dapat dilihat pada Tabel 2.3 berikut:

Tabel 2.3 *Komponen Class Diagram*

Komponen	Simbol
Class Notation, adalah sekelompok objek yang memiliki peran yang serupa dalam sistem dan terdiri dari atribut dan operasi.	
Association, mewakili hubungan antar kelas dan didefinisikan sesuai label menggunakan kata kerja.	
Generalization, menggambarkan hubungan "*is-a*" antara beberapa kelas anak yang memiliki spesialisasi dari kelas induk (warisan).	
Aggregation, hubungan "*a-part-of*" yang menunjukan bahwa satu kelas adalah bagian dari kelas lain dan memiliki *lifetime* yang berbeda.	
Composition, menunjukan hungan antar kelas yang memiliki *lifetime* yang sama dan masing-masing kelas tidak dapat berdiri sendiri	

Sumber: What is Class Diagram, (Paradigm, 2022)

2.1.5.4 Sequence Diagram

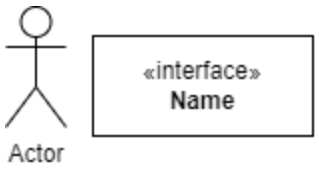
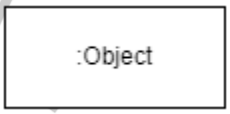
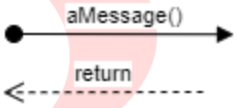
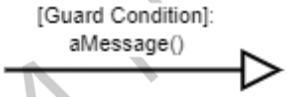
Sequence diagram adalah salah satu diagram yang memberikan gambaran detail terkait aliran kerja atau urutan proses aktivitas yang terjadi dalam sistem, proses bisnis, atau menu dalam sebuah

software. Diagram ini memvisualisasikan interaksi antara objek-objek dalam sistem secara berurutan (Syarif & Nugraha, 2020). Pengertian lainnya adalah diagram yang dapat menggambarkan perilaku suatu objek dalam suatu *use case* dengan menjelaskan masa hidup objek serta pertukaran pesan antara objek-objek tersebut. Diagram ini dibuat berdasarkan jumlah use case yang memiliki operasi khusus dan telah didefinisikan sebelumnya (Putra & Andriani, 2019).

Sedangkan menurut (Huisman & Rubin, 2017) dalam bukunya yang berjudul "*Fundamental Approaches to Software Enginering*" mendefinisikan bahwa *sequence diagram* adalah serangkaian peristiwa yang terjadi dalam perangkat lunak yang diurutkan dan dilabeli pada serangkaian proses.

Maka *sequence diagram* dapat disimpulkan sebagai sebuah diagram dalam UML yang memvisualisasikan objek-objek yang terlibat dalam sebuah use case, serta pesan-pesan yang dikirim dan diterima antara objek-objek tersebut dalam masa hidup tertentu. Komponen-komponen tersebut dapat dilihat dalam **Tabel 2.4** berikut:

Tabel 2.4 *Komponen Sequence Diagram*

Komponen	Simbol
Actor, Menggambarkan partisipasi manusia atau sistem lain dalam urutan dengan mengirim atau menerima pesan. Lokasi penempatan Aktor adalah di bagian atas diagram	
Object, adalah objek yang terlibat dalam sistem secara berurutan dengan mengirim atau menerima pesan.	
Lifeline, Sebuah garis terputus yang mengilustrasikan waktu hidup dari aktor atau objek pada sebuah sequence diagram selama urutan, *lifeline* akan ditandai dengan X pada titik dimana interaksi kelas berakhir.	
Message, Informasi atau pesan yang dikirim atau diterima antar objek dalam *sequence diagram*	
Execution Occurrence, Sebuah persegi panjang yang ditempatkan diatas *lifetime* yang menandakan waktu atau durasi terjadinya pengiriman atau penerimaan pesan.	
Guard Condition, adalah suatu persyaratan atau kondisi yang harus dipenuhi agar pesan dapat dikirimkan oleh sebuah objek atau aktor.	
Object Destruction, adalah simbol × yang ditempatkan di akhir *lifeline* untuk menunjukan ahir dari sebuah objek.	
Frame, menunjukan area atau konteks dari sequence diagram dan dapat menunjukan interaksi di dalam sequence diagram saat ada sebuah kondisional ataupun perulangan (*looping*)	

2.1.6 Pengertian Website

Website sering disebut sebagai *World Wide Web* (WWW) adalah sebuah layanan yang digunakan untuk menyajikan informasi dengan memanfaatkan prinsip *hyperlink* atau tautan yang mempermudah pengguna dalam berselancar atau mencari informasi melalui *internet*. Fungsi ini membuat web menjadi salah satu servis dengan pertumbuhan yang cepat. Melalui web, pengguna dapat menautkan kata atau gambar dalam dokumen dengan media lain seperti dokumen lain, klip video, atau file suara. Tautan web dapat dibuat dari berbagai bagian dalam dokumen atau gambar dan ditujukan ke bagian lain dalam dokumen lain. Dengan menggunakan *browser* yang dilengkapi dengan *Graphical User Interface* (GUI), pengguna dapat mengarahkan tautan ke tujuan yang diinginkan dengan mengarahkan kursor ke tautan dan menekannya (Susilo, 2018). Sedangkan pengertian *website* menurut (Brugger, 2009) mengartikan *website* sebagai serangkaian halaman yang memiliki alamat unik yang bila dimasukan kedalam aplikasi penjelajah atau *browser* akan menampilkan halaman yang dituju. *Website* diakses melalui perangkat pengguna memanfaatkan jaringan internet ataupun jaringan lainnya.

Sejalan dengan pengertian sebelumnya, (Rochman et al., 2018) menyatakan bahwa *website* adalah halaman web yang menyediakan informasi melalui internet dan bisa dibuka dari mana saja selama tersambung dengan jaringan internet. Halaman web sendiri terdiri dari beberapa bagian seperti tulisan, gambar, suara, dan animasi yang dikombinasikan sehingga menciptakan sebuah informasi yang menarik dan layak untuk dikunjungi. Ada tiga jenis *website* yang sering digunakan yaitu:

1. *Static Website*, adalah *website* yang memiliki halaman yang tidak dapat diubah (statis) membutuhkan pengeditan kode secara

langsung untuk mengubah konten halaman. Dalam hal ini, jika ada kebutuhan untuk mengubah isi dari halaman, harus dilakukan dengan cara mengubah kode yang merupakan bentuk dasar dari *website* tersebut

2. *Dynamic Website*, adalah Jenis website yang dibangun dengan struktur yang dapat menangani perubahan yang sering dan memiliki *backend* untuk mengedit konten website disebut sebagai website dinamis. Jenis website ini adalah yang paling sesuai digunakan untuk membangun aplikasi berbasis web karena memungkinkan pengelolaan dan pembaruan konten secara fleksibel melalui *backend*.
3. *Interactive Website*, adalah jenis website yang lebih mengutamakan interaksi antar pengguna, dimana setiap *user* dapat berinteraksi melalui *website* tersebut, contohnya adalah *website* media sosial ataupun portal berita yang memberikan kolom komentar untuk menyampaikan pendapat dari pembaca.

2.1.7 Pengertian PHP

Hypertext Preprocessor atau yang juga dikenal sebagai PHP adalah bahasa pemrograman skrip sisi *server* , bahasa ini bersifat terbuka atau *open source*. PHP berfungsi sebagai bahasa *scripting* yang mengeksekusi perintah pemrograman pada saat runtime. Hasil yang didapatkan dari eksekusi perintah tersebut dapat bervariasi tergantung pada data yang diproses. PHP digunakan secara khusus pada sisi *server*, di mana skrip PHP akan dijalankan dan diproses oleh *server*. Beberapa jenis *server* yang paling banyak dimanfaatkan dengan PHP antara lain Nginx, XAMPP, Apache, dan LiteSpeed (Rahmatuloh & Revanda, 2022). Pengertian lain dari PHP adalah bahasa pemrograman yang beroperasi pada *server* (*server-side script*) dan paling sering digunakan untuk mengembangkan aplikasi berbasis web. Untuk menjalankan website yang menggunakan bahasa pemrograman PHP, diperlukan *server* web yang mendukung eksekusi skrip PHP. Tanpa adanya *server web*, *website* dengan

bahasa pemrograman PHP tidak dapat berfungsi dengan baik (Anggraini et al., 2020).

Sedangkan menurut (Rochman et al., 2018) PHP adalah bahasa penulisan *script* yang pertama dikembangkan untuk menghasilkan pernyataan HTML sehingga program yang dikembangkan dengan bahasa PHP tetap memiliki tampilan layaknya HTML.

2.1.8 Pengertian Database

Database atau basis data merujuk pada gabungan data yang terorganisir secara terpusat. Istilah "basis" mengacu pada tempat atau lokasi penyimpanan, sedangkan "data" merujuk pada representasi fakta atau informasi yang berasal dari dunia nyata. Data tersebut dapat berupa angka, huruf, simbol, teks, gambar, suara, atau kombinasinya. Dalam basis data, data disimpan dalam bentuk file atau tabel terkait atau berhubungan (relasi), dan biasanya disimpan dalam alat penyimpanan elektronik (Harumy, T.H.F., Julham Sitorus, 2018). Pengertian lain dari *database* adalah kumpulan data yang terhubung satu sama lain (*interrelated data*) dan disimpan secara terpusat pada suatu media penyimpanan. Dalam basis data, tidak ada pengulangan data yang tidak terkendali (*controlled redundancy*), artinya data disimpan hanya sekali tanpa duplikasi yang tidak perlu (Ayu & Permatasari, 2018).

Berdasarkan pengertian yang telah disampaikan sebelumnya, maka basis data adalah kumpulan data yang dikelola dengan cara yang memiliki beberapa aturan khusus yang saling terkait. Melalui pengelolaan ini, pengguna dapat dengan mudah menemukan informasi, menyimpan informasi, dan mengolah informasi.

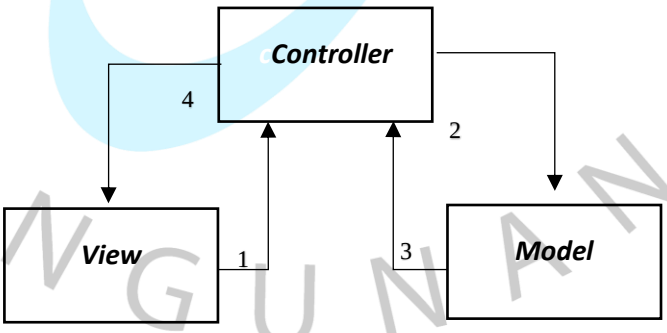
2.1.9 Pengertian MVC

MVC adalah kepanjangan dari *model view controller* adalah suatu framework yang memisahkan lapisan aplikasi-logik dari lapisan presentasi, sehingga memungkinkan pengembang untuk lebih fokus pada inti dari sistem. (Anggraini et al., 2020). MVC

ditemukan oleh Trygve Reenskaug seorang penemu Smalltalk yang Tujuannya adalah untuk menyederhanakan penulisan kode program dan menggabungkan data dengan proses pengolahan (*model*), dengan memisahkan proses pengaturan (*controller*) dan tampilan (*view*) untuk ditampilkan pada antarmuka pengguna (Rahmatuloh & Revanda, 2022). Secara singkat, MVC dapat diartikan sebagai sebuah teknik penulisan kode aplikasi dengan membagi menjadi bagian yaitu:

1. *Model* yaitu bagian yang berinteraksi dengan *database* agar dapat menyediakan, memanipulasi, dan mengorganisasikan data dalam *database.*
2. *View* adalah bagian yang berfungsi untuk menampilkan informasi yang kepada pengguna melalui Tampilan antarmuka pengguna atau GUI.
3. *Controller* merupakan komponen yang bertugas sebagai penghubung dan pengatur antara model dan view agar dapat saling berinteraksi dan terhubung secara efektif..

Alur kerja dari konsep MVC ini dapat dilihat pada **Gambar 2.3** di bawah ini.



Gambar 2.3 Alur Kerja Model-View-Controller

Proses diawali dengan *view* yang meminta data untuk ditampilkan, permintaan tersebut akan diterima oleh *controller* yang kemudian meneruskan permintaan tersebut kepada *model* untuk diproses, *model* akan mencari data yang diperlukan didalam *database* dan memprosesnya, *controller* akan mengambil data hasil

pemrosesan dan kemudian mengaturnya di bagian *view* untuk kemudian ditampilkan kepada pengguna.

2.1.10 Pengertian *Blackbox Testing*

Pengertian *Blackbox Testing* adalah sebuah metode untuk melakukan pengujian sebuah sistem informasi dari sudut pandang pengguna yang befokus pada fungsionalitas sebuah sistem tanpa memperhatikan kontrol dari sebuah sistem (Tri Snadhika Jaya, 2018). Sedangkan menurut (Yanuarsyah & Napianto, 2021) *blackbox testing* adalah sebuah metode pengujian sistem informasi yang betujuan mengetahui *input* dan *output* dari sebuah sistem tanpa memperhatikan proses yang terjadi untuk mengolah *input* menjadi sebuah *output.*

Berdasarkan pengertian diatas, maka *blackbox testing* dapat diartikan sebuah metode pengujian perangkat lunak di mana sistem diperlakukan sebagai "kotak hitam" yang tidak diketahui struktur internalnya. Pengujian dilakukan dengan fokus pada fungsionalitas sistem dan memeriksa apakah *output* yang dihasilkan sesuai dengan harapan pengguna, tanpa memperhatikan rincian implementasi internal yang terjadi untuk mengelola *intput*.

2.2. Tinjauan Studi

Dalam penelitian ini, dilakukan pengkajian terhadap berbagai jurnal yang relevan sebagai referensi dalam penyusunan laporan. Berikut ini adalah beberapa referensi yang mendukung penelitian ini.

1. Dalam penelitian yang dilakukan oleh Tirsa Ninia Lina, Matheus Supriyanto Rumetna, Frits Gerit John Rupilele, Annisa Nurul Sucianingsih Palisoa, dan Muhammad Zulkarnain Sirajjudin, yang diterbitkan di Jurnal Jendela Ilmu Sorong, Vol.1, No. 1, Tahun 2020, berjudul "**Sistem Informasi E-Arsip Berbasis Web (Studi Kasus: PT Haleyora Powerindo Cabang Sorong)**", membahas tentang masalah pengelolaan dokumen arsip di PT. Haleyora Powerindo Cabang Sorong, di mana dokumen-dokumen masih dikelola secara

manual yang menghambat aksesibilitas dokumen saat dibutuhkan dengan cepat. hasil dari penelitian ini adalah perancangan sistem informasi yang berbasis web untuk memberikan manfaat bagi perusahaan dengan meringankan beban kerja staf, meningkatkan efisiensi, dan efektivitas dalam pencarian dokumen yang diperlukan secara cepat.

2. Jurnal penelitian yang dilakukan oleh Fitri Ayu dan Nia Permatasari diterbitkan oleh Jurnal Intra-Tect Riau Vol. 2, No. 2 Tahun 2018 yang berjudul **"Perancangan Sistem Informasi Pengolahan Data Praktek Kerja Lapangan (PKL) pada Divisi Humas PT. Pegadaian".** Penelitian membahas tentang pengelolaan kegiatan Praktek Lapangan di Humas PT. Pegadaian (Persero) Kanwil II Pekanbaru dimana masih dikelola secara manual, dengan mencatat semua data PKL dan kerja sama pada buku besar sehingga manfaat perkembangan teknologi dan komunikasi dalam hal pengelolaan informasi yang lebih baik tidak sepenuhnya terpakai. Penelitian ini menghasilkan perancangan sistem informasi yang memanfaatkan bahasa PHP dan database SQL, yang secara umum dapat meningkatkan dan mempermudah proses pengelolaan data PKL, mengurangi risiko kesalahan dalam pengelolaan, dan mempercepat penyajian data PKL jika diperlukan.
3. Dalam jurnal penelitian yang dilakukan oleh Agustinus Fritz Wijaya dan Vincentius Dhendy Randya Damara, yang diterbitkan di Jurnal Bina Komputer Salatiga, Vol. 2, No. 1, Tahun 2020, dengan judul "**Perencanaan Strategis SI/TI Pada Document Management Menggunakan Ward and Peppard (Studi Kasus: PT. Visionet Data International)**", membahas tentang sistem pengelolaan dokumentasi di PT Visionet Data International yang saat ini tidak memiliki pengelolaan arsip dokumen perusahaan yang baik sesuai dengan standar ISO/IEC 20000 yang digunakan oleh perusahaan. Akibatnya, saat dilakukan review, masih banyak dokumen yang tidak diperbarui tanpa melalui proses perubahan yang terencana.

Hasil penelitian ini mencakup usulan sistem informasi yang dirancang dengan menggunakan metode Ward and Peppard, karena metode ini sangat relevan dalam perencanaan strategis sistem informasi. Metode ini menawarkan kerangka kerja yang terstruktur dan mengutamakan analisis kebutuhan bisnis daripada aspek teknologi. Beberapa teknik analisis yang digunakan dalam metode ini meliputi Analisis SWOT, Analisis Persaingan Lima Kekuatan, Analisis Rantai Nilai, Metode Faktor Sukses Kritis, Metode Balanced Scorecard, dan jaringan strategis McFarlan.

4. Dalam jurnal penelitian yang dilakukan oleh Akhmad Syafrudin dan Nur Ani, yang diterbitkan di Jurnal SISFOKOM, Vol. 08, No. 02, Tahun 2019, dengan judul "**Perancangan Sistem Informasi Pengajuan dan Pelaporan Pembayaran Tunjangan Kinerja Kementerian Keuangan Menggunakan Metode Prototype**", membahas tentang proses pengajuan dan pembuatan laporan tunjangan kerja di Biro Perencanaan dan Keuangan Kementerian Keuangan yang masih dilakukan secara manual. Proses manual ini memakan waktu yang lama, mulai dari satuan kerja sampai ke tingkat kementerian. Masalah yang sering muncul adalah kesalahan dalam melakukan input data, duplikasi data, proses rekapitulasi yang dilakukan secara berjenjang menggunakan file Excel yang rentan terjadi kesalahan input, penulisan formula, atau adanya baris atau kolom yang tersembunyi (*hide*) sehingga data yang tidak dibutuhkan masih dikutkan dalam perhitungan tunjangan.

   Selain masalah tersebut, pegawai juga mengeluhkan keterlambatan dalam pengajuan pembayaran dan penyampaian laporan pertanggungjawaban. Biro Perencanaan dan Keuangan juga menghadapi kesulitan ketika terjadi audit karena data penyetoran pajak harus dikumpulkan secara berjenjang dari satuan kerja.hasil dari penelitian ini adalah sebuah prototipe sistem informasi yang bertugas untuk mencatat pengajuan dan membuat laporan pembayaran

tunjangan kerja pada Kementerian Keuangan. Prototipe ini memiliki fitur-fitur seperti pencatatan pengajuan pembayaran dan pembuatan laporan tunjangan, pemantauan permintaan dan pelaporan tunjangan kerja, serta memberikan informasi terkait rincian penyetoran pajak.

5. Dalam sebuah penelitian yang dilakukan oleh Denny Andrian, yang diterbitkan di Jurnal Informatika dan Rekayasa Perangkat Lunak (JATIKA), Vol. 2, No. 1, Tahun 2021, dengan judul "**Penerapan Metode Waterfall Dalam Perancangan Sistem Informasi Pengawasan Proyek Berbasis Web**", membahas tentang permasalahan yang terjadi di PT. Alam Jaya Berdikari (AJB), sebuah perusahaan konstruksi. Permasalahan yang dihadapi adalah keterlambatan dalam penyampaian informasi karena lokasi proyek yang jauh, sehingga manajer proyek harus datang langsung ke lokasi proyek untuk memastikan informasi yang disampaikan oleh pelaksana proyek benar. Selain itu, penyelenggara acara juga mengalami penumpukan gambar proyek, yang menyulitkan manajer proyek untuk mengidentifikasi gambar yang sesuai dengan proyek tertentu karena gambar-gambar tersebut dikirim melalui aplikasi WhatsApp.

   Hasil dari penelitian ini adalah sebuah rancangan sistem informasi yang berfungsi untuk melakukan pelaporan perkembangan proyek dalam bentuk tabel, grafik, dan dokumentasi proyek yang sesuai dengan proyek masing-masing. Metode waterfall dengan bahasa pemrograman PHP dan database SQL digunakan untuk mengembangkan sistem informasi.
6. Dalam jurnal penelitian yang dilakukan oleh Titania Pricilia dan Zulfachmi, yang diterbitkan di Bangkit Indonesia, Vol. 10, No. 01, Tahun 2021, dengan judul "**Perbandingan Metode Pengembangan Perangkat Lunak (Waterfall, Prototype, RAD)**", penelitian ini membandingkan beberapa metode pengembangan sistem informasi, yaitu waterfall, prototype, dan rapid application development (RAD). penelitian ini bertujuan untuk memilih metode yang sesuai untuk mengembangkan sistem informasi sesuai dengan kebutuhannya.

Berdasarkan hasil penelitian ini, metode waterfall cocok digunakan untuk pengembangan sistem informasi yang bersifat umum (generic), di mana kebutuhannya sudah diketahui dari awal. Sedangkan untuk sistem atau perangkat lunak yang dibuat secara khusus sesuai dengan kebutuhan yang spesifik, metode prototype menjadi pilihan yang tepat. Sedangkan untuk pengembangan sistem dengan kebutuhan yang spesifik dan berskala besar, serta perlu diselesaikan dalam waktu singkat, metode RAD menjadi model yang sesuai.

Dengan demikian, penelitian ini memberikan pemahaman mengenai perbandingan metode pengembangan perangkat lunak dan membantu dalam pemilihan metode yang paling sesuai sesuai dengan karakteristik dan kebutuhan sistem informasi yang akan dikembangkan.

7. Dalam jurnal penelitian yang dilakukan oleh Tri Sandika Jaya dan diterbitkan di Jurnal Informatika: Jurnal Pengembangan IT (JPIT), Vol.03, No.02, Tahun 2018, dengan judul "**Pengujian Aplikasi dengan Metode Blackbox Testing Boundary Value Analysis (Studi Kasus: Kantor Digital Politeknik Negeri Lampung)**", Penelitian ini memiliki tujuan untuk melakukan pengujian terhadap aplikasi "Kantor Digital" yang digunakan oleh Politeknik Negeri Lampung. Pengujian dilakukan dengan menggunakan metode Blackbox Testing dan menerapkan teknik Boundary Value Analysis (BVA)..

   Penelitian ini menghasilkan kesimpulan bahwa metode *blackbox* testing dengan teknik BVA dapat digunakan untuk menguji fungsionalitas aplikasi "Kantor Digital". Dalam pengujian tersebut, ditemukan kekurangan pada tiga bagiab yang perlu diperbaiki guna meningkatkan performa aplikasi dalam mengelola data.

   Penelitian ini memberikan pemahaman tentang pentingnya pengujian aplikasi menggunakan metode blackbox testing dengan teknik BVA dalam konteks studi kasus "Kantor Digital" Politeknik Negeri Lampung.