BAB II
LANDASAN TEORI

2.1. Tinjauan Pustaka

Untuk membantu dalam pelaksanaan penelitian ini, dibutuhkan sebuah referensi atau literatur sebagai bahan pembelajaran, dimana literatur ini merupakan hasil-hasil mempelajari penelitian yang dilakukan oleh peneliti-peneliti dahulu yang berkaitan dengan aplikasi pengelolaan pelanggaran nilai kedisiplinan siswa berbasis *web*. Daftar literatur yang digunakan dapat dilihat pada Tabel 2.1.

Tabel 2.1 Daftar Literatur

No Literatur	Penulis	Tahun	Judul
01	Ana Qurrota A'yun, Eko Supraptono, Suryono	2016	Pengembangan Sistem *Monitoring* Pengelolaan Tata Tertib Siswa Sebagai Penunjang Layanan Bimbingan Dan Konseling
02	Endri Cahyaningrum	2016	Rancang Bangun Aplikasi Monitoring Pelanggaran Siswa Di SMK Muhammadiyah 1 Sragen
03	Damayanti & Heni Sulistiani	2017	Sistem Informasi Pembayaran Biaya Semester Pada SD Ar-Raudah Bandar Lampung
04	Reni Haerani dan Robiyanto	2019	Sistem Informasi Pengolahan Data Nilai Siswa Berbasis Web
05	Hidayati, Suhardi, Dedy Irfan, Ambiyar, Rika Melyanti	2020	Sistem Informasi Pelanggaran Siswa Berbasis Web Menggunakan Rapid Application Development

2.1.1. Tinjauan Literatur 01

Penelitian yang dilakukan oleh (A'yun, Supraptono and Suryono, 2016) dengan judul "Pengembangan Sistem *Monitoring* Pengelolaan Tata Tertib Siswa Sebagai Sarana Penunjang Penanganan Layanan Bimbingan Dan Konseling". Penelitian yang dilakukan mengangkat permasalahan *monitoring* pengelolaan tata tertib masih menggunakan pencatatan manual pada buku rekapan pelanggaran perkelas. Dikarenakan belum adanya sistem yang dapat digunakan sebagai pemantauan (*monitoring*) jumlah poin pelanggaran siswa disekolah. Hasil dari penelitian adalah SIMPETA (Sistem *Monitoring* Pengelolaan Tata Tertib Siswa) memiliki *performance* yang lebih baik dalam melakukan pengelolaan dan monitoring tata tertib siswa. Menghasilkan SIMPETA dengan penerimaan penggunaan yang baik untuk diterapkan sebagai penunjang pelayanan bimbingan dan konseling.

2.1.2. Tinjauan Literatur 02

Penelitian yang dilakukan oleh (Cahyaningrum, 2016) dengan judul "Rancang Bangun Aplikasi *Monitoring* Pelanggaran Siswa Di SMK Muhammadiyah 1 Sragen". Penelitian yang dilakukan mengangkat kasus penggunaan cara manual dalam menentukan pelanggaran tata tertib siswa, sehingga dalam pembuatan laporan masih banyak kesalahan, sehingga informasi yang dihasilkan tidak akurat dan dalam pengarsipan sering terjadi kehilangan sehingga melakukan pencarian yang membutuhkan waktu yang lama. Hasil penelitian ini adalah aplikasi monitoring pelanggaran siswa ini diharapkan dapat membantu dalam monitoring pelanggaran siswa. Guru bimbingan konseling dengan mudah mengetahui perkembangan siswa dari data pelanggaran yang telah tersistem. Siswa dapat mengakses sistem yang telah dibangun ini sebagai bahan introspeksi siswa untuk berbuat lebih baik lagi dan tidak melakukan kesalahan. Sistem informasi ini menggunakan aplikasi berbasis web yang dimana nantinya dapat memudahkan guru dan siswa dalam hal memonitoring pelanggaran.

2.1.3. Tinjauan Literatur 03

Penelitian oleh (Damayanti and Sulistiani, 2017) dengan judul "Sistem Informasi Pembayaran Biaya Sekolah Pada SD AR-Raudah Bandar Lampung". Penelitian ini mambahas pencatatan pembayaran biaya sekolah masih menggunakan buku besar yang kemudian diinputkan kedalam program Microsoft Excel sehingga sering terjadi kesalahan dalam pencatatan transaksi dan pembuatan laporan yang belum efektif dan kekurangan data. Hasil penelitan dibangun sebuah sistem informasi pembayaran biaya sekolah untuk SD AR-Raudah Bandar Lampung supaya pembuatan laporan berjalanan dengan cepat, mudah dan akurat, sistem ini juga dapat mempermudah petugas pembayaran biaya sekolah sehingga pihak pimpinan dapat dengan cepat dalam pengambilan keputusan.

2.1.4. Tinjauan Literatur 04

Penelitian yang dilakukan oleh (Haerani and Robiyanto, 2019) dengan judul "Sistem Informasi Pengolahan Data Nilai Siswa Berbasis Web". Penelitian ini membahas penilaian terhadap proses dan hasil belajar dilakukan secara manual yang diinputkan kedalam aplikasi *Report Digital* (ARD), nilai tugas, ulangan harian, ujian tengah semester dan ujian akhir semester siswa dikumpul dalam bentuk *hardcopy* yang memiliki jumlah cukup banyak dan mempersulit guru/wali kelas untuk mengontrolnya. Penginputan nilai menggunakan ARD masih menggunakan area local sehingga dapat menghambat kecepatan dalam penyampaian informasi. Dalam penelitian ini dibangun sebuah sistem pengolahan nilai yang dapat membantu kerja dari administrasi, guru/wali kelas sehingga memberikan kemudahan dalam melaksanakan proses pengolahan nilai siswa agar dapat disampaikan dengan cepat dan baik ke siswa. Sistem ini juga tidak hanya menggunakan jaringan local karena berbasis web yang dapat diakses dimana saja dengan bantuan internet, sistem juga diimplementasikan sesuai tampilan raport yang ada di sekolah sehingga sistem dapat dengan mudah menghasilkan perhitungan nilai yang akurat.

2.1.5. Tinjauan Literatur 05

Penelitian oleh (Hidayati *et al.*, 2020) dengan judul "Sistem Informasi Pelanggaran Siswa Berbasis Web Menggunakan *Rapid Application Development*". Dalam penelitian ini membahas sistem informasi pelanggaran siswa yang ada di SMA Negeri 1 Rimba masih menggunkana sistem manual atau konvensional sehingga guru Bimbingan Konseling membutuhkan waktu lama dalam penanganan siswa yang bermasalah. Pengarsipan data pelanggaran siswa disimpan dalam buku catatan secara manual sehingga terkadang hasil Analisa dan penanganan terhadap siswa menjadi tidak valid, penyampaian informasi mengenai baik buruknya perilaku siswa di sekolah kepada orangtua atau wali mengalami kesulitan. Hasil penelitian dibangun sebuah sistem berupa analisis *Rappid Aplication Development* (RAD), dengan sistem ini diharapkan sistem dapat mempermudah dan mempercepat proses penghitungan pelanggaran siswa dengan nilai-nilai point yang berlaku. Penyampaian informasi pelanggaran siswa juga dapat disampaikan dengan mudah dan cepat kepada orangtua murid / wali murid, sehingga orangtua dan guru dapat memantau tingkat kedisiplinan siswa dengan baik.

Dari beberapa literatur yang telah dibahas diatas sistem informasi dapat memberikan kemudahan pengguna sistem dalam memproses data dan memberi keputusan terhadap sesuatu hal, dan juga dapat memberikan informasi lebih cepat dan akurat. Pada penelitian yang akan dilakukan ini, peneliti akan merancang sebuah aplikasi sistem pengelolaan nilai kedisiplinan siswa yang dapat menghasilkan suatu informasi yang cepat, dan akurat. Sebuah perbedaan penulis dengan peneliti terdahulu yaitu penulis menggunakan pemograman *codeigneter*, metode yang digunakan metode *extreme programming* dan pengujian kualitasnya memakai *blackbox testing.*

2.2. Kedisiplinan

Menurut Rahman dalam jurnal Nugroho dan Sami'a (2016) mengatakan bahwa disiplin berasal dari bahasa Inggris "discipline" yang mengandung beberapa arti pengendalian diri, membentuk karakter yang bermoral,

memperbaiki dengan sanksi, serta kumpulan beberapa tata tertib untuk mengatur tingkah laku.

Disiplin siswa berarti dan ketaatan siswa terhadap berbagai aturan dan tata tertib yang berlaku disekolahnya. Disiplin merupakan kesediaan untuk memenuhi peraturan dan larang-larangan. Kepatuhan disini bukan hanya patuh karena adanya tekanan-tekanan dari luar melainkan kepatuhan yang didasari adanya kesadaran tentang nilai dan pentingnya peraturan-peraturan dan larangan-larangan tersebut.

2.3. *Web*

Menurut (Agus Hariyanto, 2015), *website* adalah *web* dapat diartikan sebagai kumpulan halaman yang menampilkan informasi data teks, data gambar, data animasi, suara, video dan gabungan semuanya, baik yang bersifat statis maupun dinamis yang membentuk suatu rangkaian bangunan yang saling terkait, dimana masing-masing dihubungkan dengan jaringan-jaringan halaman (*hyperlink).*

Sedangkan menurut (Rohi abdulloh, 2015), *web* adalah sekumpulan halaman yang terdiri dari beberapa halaman yang berisi informasi dalam bentuk data digital baik berupa teks, gambar, video, audio, dan animasi lainnya yang disediakan melalui jalur koneksi internet.

2.4. *Database MySQL*

Menurut (MF, 2018) mengatakan bahwa MySQL merupakan sistem manajemen database SQL yang bersifat *opensource* dan paling banyak digunakan saat ini. Sistem database MySQL mampu mendukung beberapa fitur seperti *multithreaded, multi-user,* dan SQL *database management sistem* (DBMS).

Open source menyatakan bahwa *software* ini dilengkapi dengan *source code* (kode yang dipakai untuk membuat *MySQL),* selain itu tentu saja bentuk *execuble-nya* atau kode yang dapat dijalankan secara langsung dalam sistem operasi.

2.5. *Framework*

Menurut (Naista, 2017) framework merupakan suatu struktur konseptual dasar digunakan untuk memecahkan atau menangani suatu masalah yang bersifat kompleks, framework merupakan suatu kerangka kerja dari sebuah website yang akan dibangun. Sedangkan menurut (Raharjo, 2018) *framework* adalah desain struktur dasar yang dapat digunakan kembali (*reusable)* yang terdiri dari *abstract class* dan *concrete class* di pemograman yang berorientasi objek.

Menurut (Irwansyah et al., 2018) keuntungan yang diperoleh dari penggunaan *framework* adalah:

1. Waktu pembuatan aplikasi website jauh lebih singkat.
2. Kode aplikasi *website* menjadi lebih mudah dibaca, karena sedikit dan sifatnya pokok, detailnya adalah kode dari *framework*.
3. *Website* menjadi lebih mudah diperbaiki, karena tidak perlu fokus kesemua komponen kode *website*, terutama kode sistem *framework*.
4. Tidak perlu lagi membuat kode penunjang aplikasi *website* seperti koneksi *database, validasi form, GUI,* dan keamanan.
5. Pikiran pengembang menjadi lebih terfokus ke kode alur permasalahan *website*, apa yang ditampilkan dan layanan apa saja yang diberikan dari aplikasi *web* tersebut.
6. Jika dikerjakan *team-work* maka akan lebih terarah karena sistem *framework* mengharuskan adanya keteraturan peletakan kode. Seperti bagian pengambilan *database* terpisah dengan bagian pengaturan tampilan untuk pengunjung.

2.6. *Codeigneter*

Menurut (Budiyanto and Surya, 2018) *CodeIgniter* yaitu sebuah framework untuk membuat website yang bersifat *opensource*. Framework ini menggunakan sistem MVC (*Model, View, Controller)* yang dikembangkan oleh Rick Ellis pada tahun 22006 untuk membangun website dinamis dengan menggunakan PHP juga mempercepat pengembang dalam melakukan pembuatan aplikasi website. Selain ringan dan cepat, *CodeIgniter* juga

memiliki dokumentasi yang lengkap karena itu menjadi salah satu alasan kenapa banyak digunakan di seluruh dunia.

2.7. *XAMPP*

Xampp yaitu sebuah paket kumpulan *software* yang terdiri dari *Apache, MySQL, PhpMyAdmin, PHP, Perl, Filezilla,* dan lain. Xampp berfungsi untuk memudahkan instalasi lingkungan PHP, dimana biasanya lingkungan pengembangan web memerlukan PHP, Apache, MySQL dan PhpMyAdmin (MADCOM, 2016).

Menurut (Haqi and Setiawan, 2019) Xampp merupakan perangkat lunak bebas *(free softare)* yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. Xampp juga merupakan sebuah aplikasi perangkat lunak pemrograman dan database yang di dalamnya terdapat berbagai macam aplikasi pemrograman seperti : Apache, HTTP, MySQL, database, bahasa pemrograman PHP dan Perl (Aryanto, 2019).

2.8. *PHP*

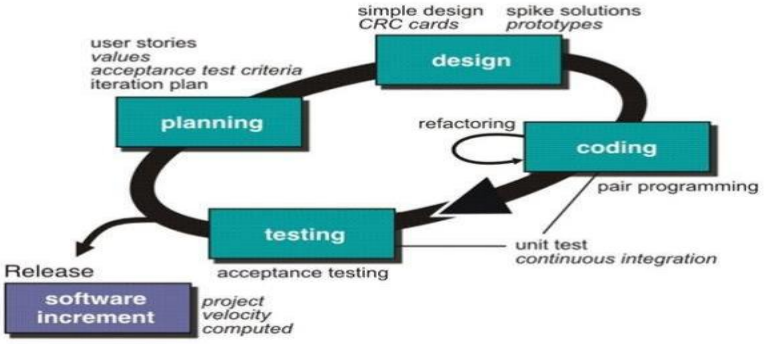
PHP (*Hypertext Preprocessor*) adalah *script* bersifat *server-side* yang ditambahkan ke dalam HTML. PHP sendiri merupakan singkatan *PersonalHome PageTools*. Skrip ini akan membuat suatu aplikasi dapat diintegrasikan ke dalam *HTML* sehingga suatu halaman *web* tidak lagi bersifat statis, namun menjadi bersifat dinamis. Sifat *serverside* berarti pengerjaan skrip dilakukan di *server*, baru kemudian hasilnya dikirim ke *browser*. (Parlika, Mubarok and Munir, 2017).

Menurut (MADCOM, 2016) PHP *(Hypertext Preprocessor)* merupakan bahasa *script* yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk membuat program situs web dinamis". PHP dapat digunakan dengan gratis dan bersifat *Open Source*.

2.9. Metode *Extreme Programming*

Menurut (Supriyatna, 2018) *Extreme programming (XP)* merupakan sebuah proses rekayasa perangkat lunak yang cenderung menggunakan pendekatan berorientasi objek dan sasaran dari metode ini adalah tim yang

dibentuk dalam skala kecil sampai medium serta metode ini juga sesuai jika tim dihadapkan dengan *requirement* yang tidak jelas maupun terjadi perubahan-perubahan *requirement* yang sangat cepat. Adapun tahapan dalam *extreme programming* dapat dilihat pada gambar 2.1.



Gambar 2.1 *Extreme Programming*

Penjelasan tahapan *extreme programming* yaitu :

1. *Planning* (perencanaan)

   Tahapan ini dimulai dengan merencanakan kebutuhan aktifitas suatu sistem yang akan dibangun yang memungkinkan pengguna memahami proses suatu sistem dan mendapatkan gambaran yang jelas mengenai fitur utama, fungsionalitas keluaran yang diinginkan, *user stories, values*.

2. *Design* (perancangan)

   Pada tahapan perencangan dilakukan pembuatan pemodelan sistem berdasarkan hasil analisa kebutuhan yang didapatkan. Pemodelan sistem dengan desain CRC *Card*, perancangan sistem menggunakan *Unified Modelling Language (UML)* dan juga dilakukan perancangan tampilan sistem *(interface)* yang akan dibangun.

3. *Coding* (pengkodean)

   Pada tahapan ini merupakan implementasi dari perancangan model sistem yang telah dibuat kedalam kode program yang menghasilkan *prototype* dari perangkat lunak.

4. *Testing* (pengujian)

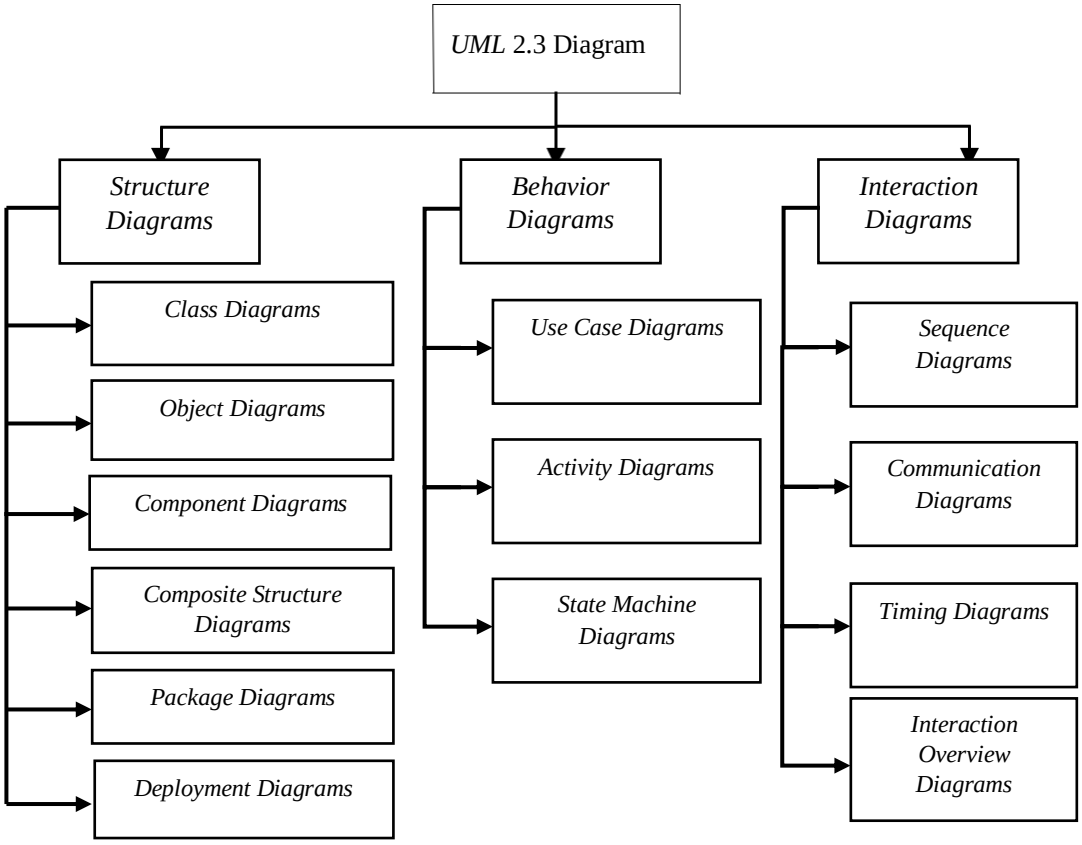
   Pada tahapan ini merupakan tahapan pengujian terhadap sistem yang sudah dibangun, pada tahapan ini ditentukan oleh pengguna sistem berfokus pada fitur dan fungsionalitas dari keseluruhan sistem kemudian ditinjau oleh pengguna sistem.

5. *Software Increment* (peningkatan perangkat lunak)

   Pada tahapan ini merupakan tahap pengembangan sistem yang sudah dibuat secara bertahap yang dilakukan setelah sistem diterapkan dalam organisasi dengan menambahkan layanan atau konten yang mengakibatkan bertambahnya kemampuan fungsionalitas dari sistem.

2.10. *Unified Modelling Language (UML)*

Menurut (A.S and Salahuddin, 2018), *UML (Unified Modelling Language)* adalah standar bahasa yang banyak digunakan didunia industri untuk mendefinisikan *requirement*, membuat analisis dan *design*, serta menggambarkan arsitektur dalam pemograman berorientasi objek. Pada *UML* 2.3 terdiri dari 13 macam diagram yang dapat dikelompokkan dalam 3 kategori. Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar 2.2.



Gambar2.2 *Diagram UML*

Sumber: (A.S and Salahuddin, 2018)

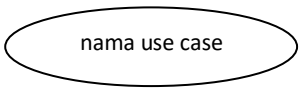
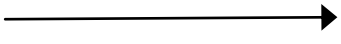
Berikut penjelasan singkat dari pembagian kategori tersebut:

1. *Structure Diagrams*, yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2. *Behavior Diagrams*, yaitu kumpulan diagram yang digunakan untuk menggambar kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
3. *Interaction Diagrams*, yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

2.10.1. *Use Case Diagram*

Menurut (A.S and Salahuddin, 2018) *Use Case diagram* merupakan pemodelan untuk mendeskripsikan sebuah interaksi antara satu atau lebih *actor* dengan sistem informasi yang akan dibuat. Secara garis besar, use case digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.Gambaran fungsi *use case* pada tabel 2.2.

Tabel 2.2 Simbol-Simbol Use Case Diagram

Simbol	Keterangan
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antara unit atau actor, biasanya menggunakan kata kerja.
Include <<*include*>> 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankannya *use case* ini
Generalisasi 	Relasi antar kelas dengan makna generalisasi-generalisasi (umum khusus)
Asosiasi/*Association* 	Komunikasi antara aktor dan *usecase* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.

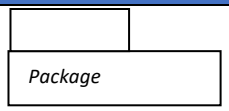
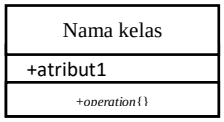
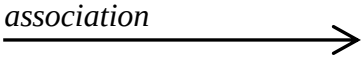
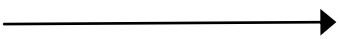
Sumber: (A.S and Salahuddin, 2018)

2.10.2. *Class Diagram*

Menurut (A.S and Salahuddin, 2018) *class diagram* merupakan gambaran dari struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas berikut:

1. Kelas yang menangani tampilan sistem, yaitu kelas mendefinisikan dan mengatur tampilan ke pemakai.
2. Kelas yang diambil dari pendefinisian *use case*, yaitu kelas menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case*.
3. kelas yang diambil dari pendefinisian data, yaitu kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan kebasis data.

Tabel 2.3 Simbol-Simbol Class Diagram

Simbol	Keterangan
Package 	*Package* merupakan contoh sebuah bungkusan dari satu atau lebih kelas
Operasi 	Kelas pada struktur *sistem*
Antarmuka/*interface* 	Sama dengan konsep *interface* dalam pemograman berorientasi objek
Asosiasi/*Association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Keberangkatan/*dependency*	Relasi antar kelas dengan makna kebergantungan antara kelas
Agregasi/*aggregation*	Relasi antar kelas dengan makna semua bagian (*whole-part*)
Asosiasi berarah/*directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antar kelas dengan makna generalisasi (umum khusus)

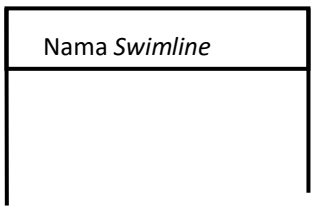
Sumber: (A.S. and Shalahuddin, 2018)

2.10.3. *Activity Diagram*

Menurut (A.S. and Shalahuddin, 2018), *activity diagram* adalah diagram aktivitas yang menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
2. Urutan atau pengelompokan tampilan dari sistem atau *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan.

Tabel 2.4 Simbol-Simbol *Activiy Diagram*

Simbol	Keterangan
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan/*decision* 	Asosiai percabangan di mana jika ada pilihan aktivitas lebih dari satu
Penggabungan/*join* 	Asosisasi penggabungan dimana lebih dari satu aktivitas digambarkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Sumber: (A.S. and Shalahuddin, 2018)

2.11. *Blackbox Testing*

Menurut (Santoso et al., 2019), *Blackbox Testing* adalah metode pengujian perangkat lunak dimana struktur atau desain internal sistem tidak diketaui oleh penguji. Prosedur tersebut didasarkan pada analisis spesifikasi, fungsional, dan nonfungsional dari suatu komponen sistem dengan tidak melihat struktur kode dan informasi internal sistem. Pada tes ini, peneliti hanya berfokus menguji *input* dan *ouput* dari sistem saja. *Input* dan *output* tersebut dinyatakan valid atau tidak valid dan sesuai atau tidak sesuai. Tujuan utama dari *blackbox testing* adalah memeriksa apakah perangkat lunak bekerja sesuai persyaratan atau kebutuhan pengguna atau tidak.

Menurut (A.S and Salahuddin, 2018) *blackbox testing* adalah menguji perangkat lunak dari spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian blackbox testing mencoba untuk menemukan kesalahan dalam beberapa kategori berikut: (1) Fungsi yang hilang atau tidak benar; (2) Kesalahan antarmuka; (3) Kesalahan pada struktur data atau pengaksesan *external database*; (4) Kesalahan perilaku atau kinerja; (5) Kesalahan Inisialisasi.