BAB II

TINJAUAN PUSTAKA

2.1 Teori Umum

2.1.1 Pengertian Perangkat Lunak

Sukamto dan Shalahuddin (2018:2), "Perangkat Lunak (*Software)* adalah program komputer yang terasosiasi dengan dokumentasi perangkat lunak seperti dokumentasi kebutuhan, model desain, dan cara penggunaan (*user manual*)".

Sedangkan Swara dan Pebriadi (2016:28), "Perangkat Lunak merupakan seluruh perintah yang digunakan untuk memproses informasi. Perangkat lunak dapat berupa program maupun prosedur yang didalam nya merupakan kumpulan perintah yang dimengerti oleh computer".

Dari kedua pendapat diatas dapat disimpulkan bahwa pengertian perangkat lunak adalah program komputer yang digunakan untuk memproses informasi dan terasosiasi dengan dokumentasi yang didalamnya terdapat kumpulan perintah yang dimengerti oleh komputer.

2.1.2 Pengertian Komputer

Kadir dalam Faunisah dkk. (2019:23) mengemukakan bahwa, "Komputer merupakan peralatan elektronik yang bermanfaat ,melaksanakan berbagai pekerjaan yang dilakukan oleh manusia ".

Hanafri dkk. (2019:88) menyatakan bahwa, "Komputer merupakan alat untuk mengolah data sesuai perintah yang sudah dirumuskan secara cepat dan tepat, serta diorganisasikan supaya secara otomatis menerima dan menyimpan data berdasarkan intruksi intruksi yang telah tersimpan didalam memori"

Berdasarkan kedua pendapat diatas dapat disimpulkan bahwa komputer adalah peralatan elektronik untuk mengolah data sesuai perintah secara otomatis berdasarkan intruksi intruksi yang telah disimpan oleh memori.

2.1.3 Pengertian Aplikasi

Supadi dalam Nurhayati dkk. (2017 :15) Aplikasi adalah salah satu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas.

Budiharto dalam Faunisah dkk. (2019:34) "Aplikasi merupakan program yang dapat berjalan di komputer tersendiri (*Stand alone computer*), dari mulai program yang simple sampai dengan program besar dan rumit".

Berdasarkan kedua pendapat diatas dapat disimpulkan bahwa aplikasi adalah program yang dapat berjalan di komputer tersendiri yang dibuat untuk melayani kebutuhan akan beberapa aktivitas.

2.1.4 Metode Pengembangan Sistem

Penelitian ini menggunakan metode pengembangan perangkat lunak dengan RUP (Rational Unified Process). Sukamto dan Shalahuddin (2016:125), "RUP (Rational Unified Process) adalah pendekatan pengembangan perangkat lunak yang dilakukan berulang-ulang (iterative), fokus pada arsitektur (architecture-centric), lebih diarahkan berdasarkan penggunaan kasus (use case driven)". Adapun tahap-tahap (fase) dalam metode pengembangan RUP menurut Sukamto dan Shalahuddin (2016:128-131) adalah sebagai berikut:

1. *Inception* (permulaan)

   Tahap ini lebih pada memodelkan proses bisnis yang dibutuhkan (bussiness modeling) dan mendefinisikan kebutuhan akan sistem yang akan dibuat (requirements).

   Pada fase ini dapat disimpulkan adanya kegiatan yang perlu dilakukan yaitu studi kelayakan dan kebutuhan fungsional dan non fungsional.

   A. Penyelidikan menurut Yahaya dkk. (2006:2) pada hakikatnya adalah bagi mencari jawapan atau menyelesaikan masalah dengan jawapan yang benar atau sekurang – kurangnya dapat mengetahui kebenaran yang logic melalui pemikiran yang mempunyai kesinambungan dengan fakta empiric.

   B. Studi kelayakan (Feasibility study) menurut Jogiyanto dalam Syaifullah & Widianto (2014:201) adalah suatu studi yang akan digunakan untuk menentukan kemungkinan apakah pengembangan proyek sistem layak

diteruskan atau dihentikan. Studi kelayakan disebut juga dengan istilah High point review.

Studi Kelayakan berisi :

1. Kelayakan Teknis menurut Al Fatta dalam Syaifullah & Widianto (2014:201) Kelayakan teknis menyoroti kebutuhan sistem yang telah disusun dari aspek teknologi yang akan digunakan, jika teknologi yang dikehendaki untuk pengembangan sistem merupakan teknologi yang mudah didapat, murah, dan tingkat pemakaiannya mudah, maka secara teknis usulan kebutuhan sistem bisa dinyatakan layak.
2. Kelayakan Operasional menurut Jogiyanto dalam Syaifullah & Widianto (2014:202) merupakan Penilaian terhadap kelayakan operasional digunakan untuk mengukur apakah sistem yang akan dikembangkan nantinya dapat dioperasikan dengan baik atau tidak di dalam orgnisasi.
3. Kelayakan Ekonomi menurut Al Fatta dalam Syaifullah & Widianto (2014:201) Tidak dapat disangkal lagi, motivasi pengembangan sistem informasi pada perusahaan atau organisasi adalah motif keuntungan.Dengan demikian aspek untung rugi jadi pertimbangan utama dalam pengembangan sistem. Kelayakan ekonomi berhubungan dengan return investmen atau berapa lama biaya investasi dapat kembali.

C. Kebutuhan fungsional menurut Wijaya (2017 : 81) merupakan jenis kebutuhan yang berisi proses-proses apa saja yang mampu dilakukan oleh sistem beserta informasi-informasi yang dihasilkan oleh sistem. Lalu Kebutuhan Non Fungsional (NFRs) menurut Suharso (2015:1) berkaitan erat dengan kebutuhan kualitas perangkat lunak. Karena semua karakteristik kebutuhan kualitas perangkat lunak adalah NFRs. Dalam prakteknya, NFRs sulit diidentifikasi karena dinyatakan dalam spesifikasi tekstual.

2. *Elaboration* (perluasan/perencanaan)

Tahap ini lebih difokuskan pada perencanaan arsitektur sistem. Tahap ini juga dapat mendeteksi apakah arsitektur sistem yang diinginkan dapat dibuat atau tidak. Mendeteksi resiko yang mungkin terjadi dari arsitektur yang dibuat.

Tahap ini lebih pada analisis dan desain sistem serta implementasi sistem yang fokus pada purwarupa sistem (prototype).

3. *Construction* (kontruksi)

   Tahap ini fokus pada pengembangan komponen dan fitur-fitur sistem. Tahap ini lebih pada implementasi dan pengujian sistem yang fokus pada implementasi perangkat lunak pada kode program. Tahap ini menghasilkan produk perangkat lunak dimana menjadi syarat dari Initial Operational Capability Milestone atau batas/tonggak kemampuan operasional awal.

   Pada fase ini terdapat kegiatan yang perlu dilakukan yaitu uji aplikasi dengan menggunakan Black-Box Testing menurut Khan (dalam Mustaqbal dkk. 2015:33) merupakan pengujian yang berfokus pada spesifikasi fungsional dari perangkat lunak, tester dapat mendefinisikan kumpulan kondisi input dan melakukan pengetesan pada spesifikasi fungsional program.

4. *Transition* (transisi)

   Tahap ini lebih pada deployment atau instalasi sistem agar dapat dimengerti oleh user. Tahap ini menghasilkan produk perangkat lunak dimana menjadi syarat dari Initial Operational Capability Milestone atau batas/tongggak kemampuan operasional awal. Aktifitas pada tahap ini termasuk pada pelatihan user, pemeliharaan dan pengujian sistem apakah sudah memenuhi harapan user.

2.2 Teori Khusus

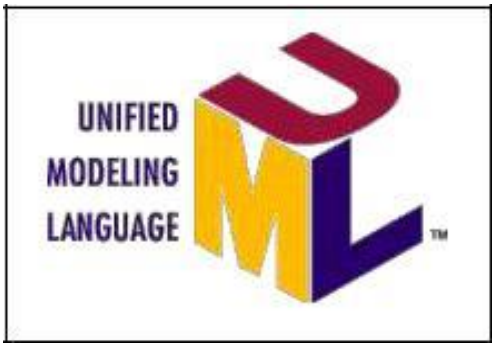
2.2.1 Pengertian *Unified Modeling Language* (UML)

Sukamto dan Shalahuddin (2018:133), menjelaskan tentang pengertian *Unified Modeling Language* sebagai berikut :

"Unified Modeling Language (UML) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan requirement, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek."

UML menyediakan serangkaian gambar dan diagram yang sangat baik. Beberapa diagram memfokuskan diri pada ketangguhan teori *object-oriented* dan

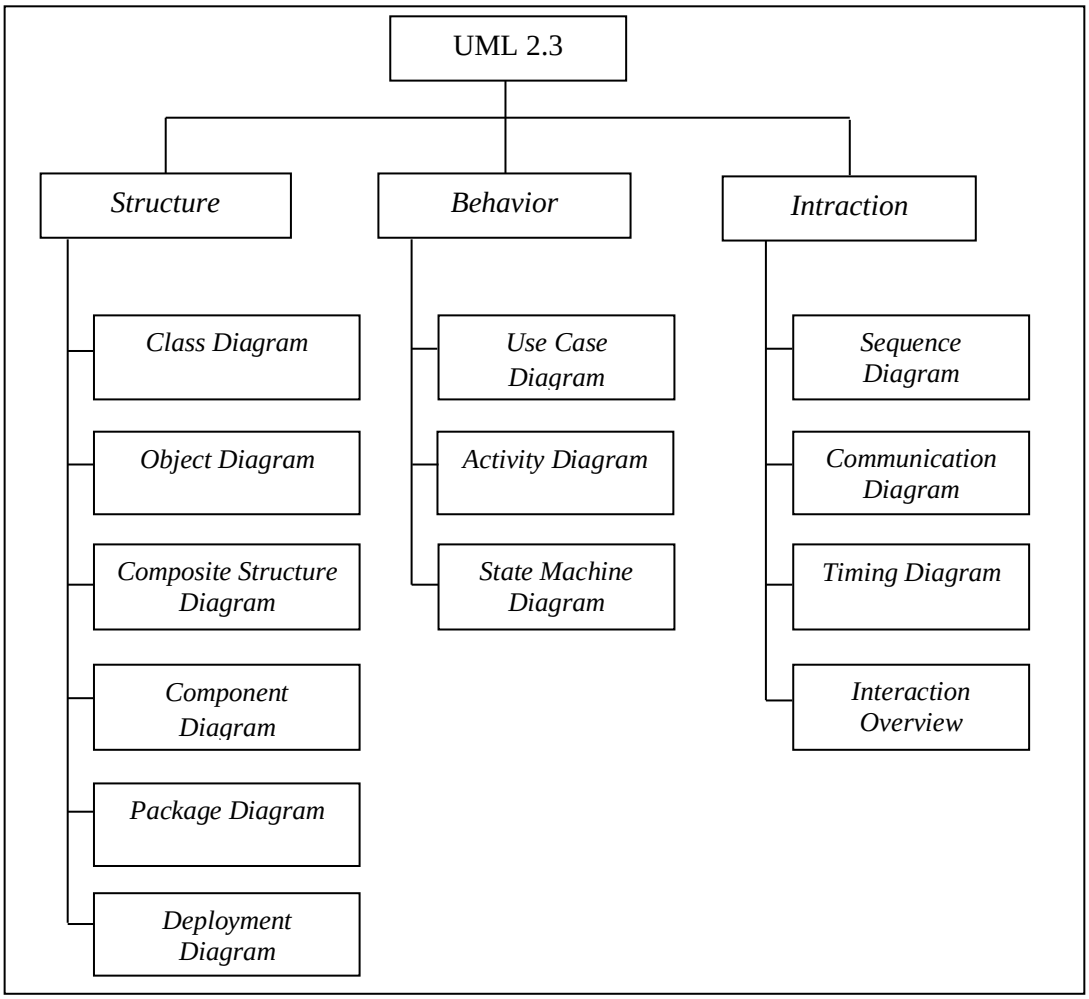
sebagian lagi memfokuskan pada detail rancangan dan konstruksi. Semua dimaksudkan sebagai sarana komunikasi antar *team programmer* maupun dengan pengguna.



Gambar 2.1. Tampilan Logo UML

2.2.2 Kategori *Unified Modeling Language* (UML)

Menurut Sukamto dan Shalahuddin (2018:140), "Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori". Pembagian kategori dan macam-macam diagram Menurut Sukamto dan Shalahuddin tersebut dapat dilihat pada gambar dibawah:



Gambar 2.2 Kategori dan Macam-macam Diagram UML

Penjelasan singkat dari pembagian kategori pada diagram UML menurut Sukamto dan Shalahuddin (2018:141) :

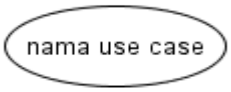
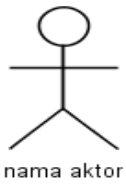
1) *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2) *Behavior diagrams* yaitu kumpulam diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahana yang terjadi pada sebuah sistem.
3) *Interaction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

2.2.3 Jenis-Jenis Diagram UML

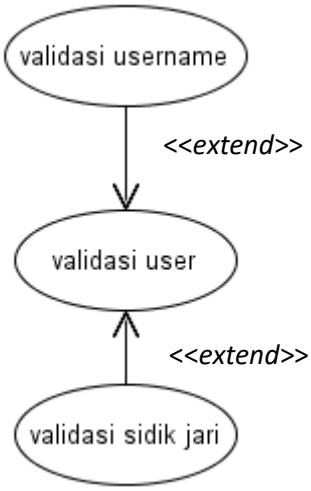
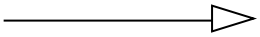
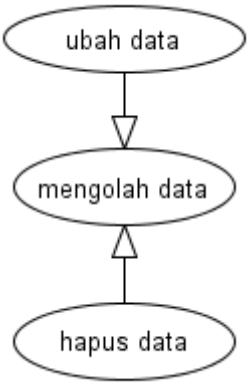
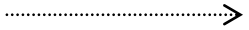
2.2.3.1 Pengertian *Use case* Diagram

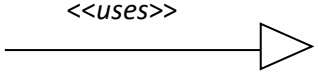
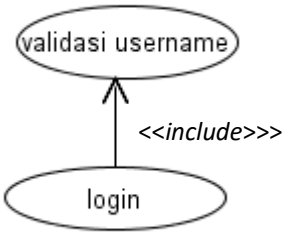
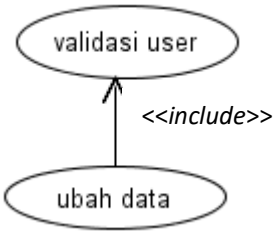
Menurut Sukamto dan Shalahuddin (2018:155), *"Use case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem." Adapun simbol-simbol yang digunakan dalam *use case* adalah sebagai berikut:

Tabel 2.1 Simbol-simbol *Use case* Diagram

No	Simbol	Deskripsi
1.	*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-awal frase nama *use case.*
2.	aktor / *actor* 	orang, proses, atau siatem 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 actor.
3.	asosiasi / *association* _____	komunikasi antar aktor dan *use case* yang berpartisioasi pada *use case* atau *use case* memiliki ingeraksi dengan actor.
4.	ekstensi / *extend* 	relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang di tambahkan 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, misalnya

Lanjutan **Tabel 2.1 Simbol-simbol *Use case* Diagram**

No	Simbol	Deskripsi
		 arah panah mengarah pada *use case* yang ditambahkan; biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya
5.	Generalisasi / *generalization* 	hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya,  misalnya: arah panah mengarah pada *use case* yang menjadi generalisasinya (umum).
.	menggunakan / include / uses  <<*include*>>	relasi tambahan ke sebuah *use case* dimana *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*:

No	Simbol	Deskripsi
		1. *Include* berarti *use case* yang ditambahkan akan selalu di panggil saat *use case* tambahan dijalankan, misalnya pada kasus berikut:  2. *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang di tambahkan telah dijalankan sebelum *use case* tambahan dijalankan, misal pada kasus berikut:  kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

Sumber: Sukamto dan Shalahuddin (2018:156)

Ada dua hal utama pada *use case* yaitu:

1. Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, Jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang.
2. *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

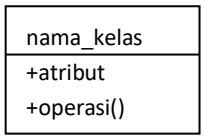
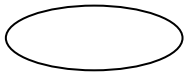
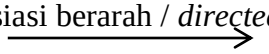
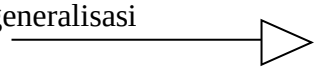
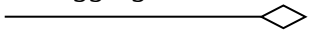
2.2.3.2 Pengertian *Class Diagram*

Sukamto dan Shalahuddin (2018:141), *"Class Diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk

membangun sistem. Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron."

Adapun simbol-simbol yang digunakan dalam *class* diagram adalah sebagai berikut:

Tabel 2.2 Simbol-simbol *Class* Diagram

No	Simbol	Deskripsi
1.	kelas 	Kelas pada struktur sistem
2	antarmuka / interface  nama_interface	Sama dengan konsep interface dalam pemrograman berorientasi objek
3.	asosiasi / association 	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai *multiplicity*
4.	asosiasi berarah / *directed assosiation* 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
5.	generalisasi 	Relasi antarkelas dengan makna generalisasi – spesialisasi (umum khusus)
6.	kebergantungan / *dependency* 	Relasi antarkelas dengan makna kebergantungan antar kelas
7.	agregasi / *aggregation* 	Relasi antarkelas dengan makna semua-bagian (*whole-part*)

Sumber: Sukamto dan Shalahuddin

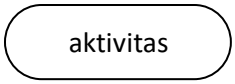
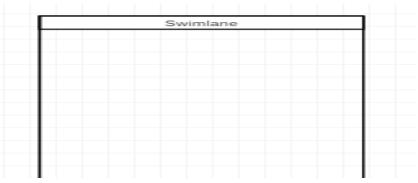
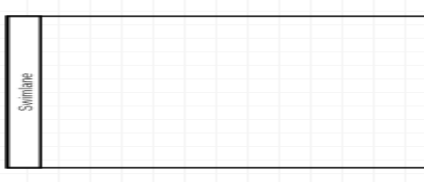
2.2.3.3 Pengertian *Activity Diagram*

Menurut Sukamto dan Shalahuddin (2018:161), "A*ctivity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan

disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem."

Adapun simbol-simbol yang digunakan dalam *activity diagram* adalah sebagai berikut:

Tabel 2.3 Simbol-simbol *Activity Diagram*

No	Simbol	Deskripsi
1.	Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
2.	Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
3.	Percabangan / *decision* 	Asosiasi percabangan di mana jika ada pilihan aktivitas lebih dari satu
4.	Penggabungan / *join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
5.	Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
6.	Swimlane  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

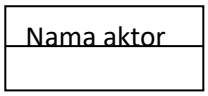
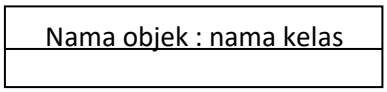
Sumber: Sukamto dan Shalahuddin

2.2.3.4 Pengertian *Sequence Diagram*

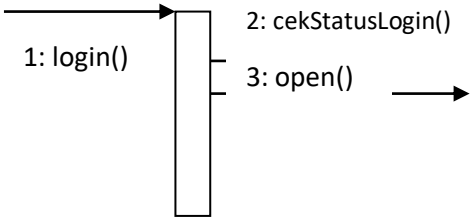
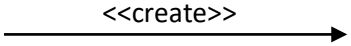
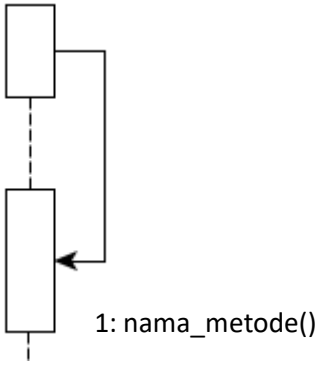
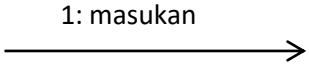
"Diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah beserta metode-metode yang dimiliki kelas yang diinstansikan menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada use case" (Sukamto dan Shalahuddin, 2018:165).

Berikut simbol-simbol pada Sequence Diagram :

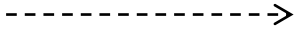
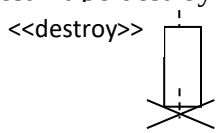
Tabel 2.4 Simbol-simbol pada Sequence Diagram

No	Simbol	Deskripsi
1.	Actor  atau  tanpa waktu aktif	orang, proses, atau sistem lain yang berinteraksi dengan 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 actor
2.	Garis hidup / *lifeline* 	menyatakan kehidupan suatu objek
3.	Objek  	menyatakan objek yang berinteraksi pesan

Lanjutan **Tabel 2.4** Simbol-**simbol** pada Sequence Diagram

No	Simbol	Deskripsi
4.	Waktu aktif 	menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya  maka cek Status Login () dan open() dilakukan di dalam metode login() aktor tidak memiliki waktu aktif
5.	Pesan tipe create 	menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
6.	Pesan tipe call	menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri,  arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi
7.	Pesan tipe send 	menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi

Lanjutan **Tabel 2.4** Simbol-**simbol** pada Sequence Diagram

No	Simbol	Deskripsi
8.	Pesan tipe return 1: keluaran 	menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian
9.	Pesan tipe destroy 	menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy

Sumber: Sukamto dan Shalahuddin (2018:165-167)

2.3 Teori Judul

2.3.1 Pengertian Pencatatan

Stice dan Skousen dalam Faunisah dkk. (2019:34) yang dimaksud "pencatatan adalah laporan keuangan yang akurat dapat dihasilkan jika hasil peristiwa dan aktivitas bisnis telah direkam atau dicatat dengan tepat".

Mulyadi dalam Utomo dkk. (2018:21) mendefinisikan , "Pencatatan adalah suatu urutan ketiga klerikal biasanya melibatkan beberapa orang dalam suatu departemen atau lebih yang dibuat untuk menjamin penanganan secara seragam terhadap transaksi perusahaan yang terjadi berulangulang"

Berdasarkan kedua pendapat diatas dapat ditarik kesimpulan bahwa pengertian dari pencatatan adalah suatu kegiatan yang melibatkan beberapa orang dalam suatu departemen atau lebih untuk menjamin penanganan transaksi perusahaan agar dapat menghasilkan laporan keuangan yang akurat.

2.3.2 Pengertian Pembelian

Mulyadi dalam Solihin dan Nusa (2017 : 108), "Pembelian adalah sebagai salah satu fungsi dari pembelanjaan atau merupakan kegiatan dari pembelanjaan. Pembelian sama pentingnya dengan penjualan, yaitu untuk memenuhi kebutuhan

setiap perusahaan, seperti kebutuhan peralatan kantor, gedung, peralatan produksi, dan lain sebagainya."

Nugroho dalam Nurhayati dkk. (2017 :16), "Pembelian adalah transaksi belanja untuk barang masuk atau pengeluaran uang yang kita lakukan untuk mendapatkan produk yang akan dijual, transaksi ini terjadi pada supplier yang produknya dibeli".

Dari kedua pendapat diatas dapat disimpulkan bahwa pengertian pembelian adalah salah satu fungsi pembelanjaan dimana terdapat transaksi untuk barang masuk dan pengeluaran uang yang dilakukan untuk mendapatkan produk.

2.3.3 Pengertian Kesehatan dan Keselamatan Kerja (K3)

Wahyudi (2018:473) Definisi Kesehatan dan Keselamatan Kerja adalah bidang yang terkait dengan kesehatan, keselamatan, dan kesejahteraan manusia yang bekerja disebuah institusi maupun lokasi proyek.

Mangkunegara dalam Wahyudi (2018:473) "Kesehatan dan Keselamatan kerja adalah suatu pemikiran dan upaya untuk menjamin keutuhan dan kesempurnaan baik jasmaniah maupun rohaniah tenaga kerja pada khususnya, dan manusia pada umumnya, hasil karya dan budaya untuk menuju masyarakat adil dan Makmur".

Dari kedua pendapat diatas dapat disimpulkan bahwa pengertian kesehatan dan keselamatan kerja adalah pemikiran dan upaya yang terkait dengan kesehatan, keselamatan, dan kesejahteraan manusia yang diselenggarakan untuk menjamin tenaga kerja dan mewujudkan produktivitas kerja secara optimal.

2.3.4 Pengertian Aplikasi Pencatatan Pembelian Barang Kesehatan dan Keselamatan Kerja pada PT. Meraksa Raya Group Bebasis Website

Pengertian Aplikasi Pencatatan Pembelian Barang Kesehatan dan Keselamatan Kerja pada PT. Meraksa Raya Group Bebasis Website adalah Sebuah aplikasi untuk mempermudah pegawai dalam pencatatan dan pengolahan data pembelian barang kesehatan dan keselamatan kerja di PT. Meraksa Raya Group.

2.4 Teori Program

2.4.1 Pengertian Basis Data

Swara dan Pebriadi (2016:29) mengemukakan bahwa “database merupakan sekumpulan table, hubungan dan lain-lain yang berkaitan dengan penyimpanan data”.

2.4.2 Pengertian *MySQL*

Sukamto dalam Faunisah dkk. (2019:36) “MySQL adalah bahasa yang digunakan untuk mengelola data pada *Relation Database Management System* (RDBMS) yang dikembangkan berdasarkan teori aljabar relasional dan kalkulus “.

2.4.3 Pengertian Xampp

Riyanto dalam Faunisah dkk. (2019:36) “Xampp adalah paket PHP dan Mysql berbasis open source, yang dapat digunakan sebagai tool pembantu pengembangan aplikasi berbasis PHP”.