BAB II

LANDASAN TEORI

2..1. Konsep Dasar Sistem

Secara umum sistem dapat diartikan sebagai kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu sebagai satu kesatuan. Menjelaskan teori yang berhubungan dengan Tugas Akhir ini , penjelasannya sebagai berikut :

2.1.1. Pengertian Sistem

Menurut Pratama (2014:7) “Sistem didefinisikan sebagai sekumpulan prosedur yang saling berkaitan dan saling terhubung untuk melakukan suatu tugas bersama-sama”

Sedangkan menurut Tohari (2014:2) “Sistem adalah kumpulan atau himpunan dari unsur atau variable yang saling terkait, saling berinteraksi, dan saling tergantung satu sama lain untuk mencapai tujuan”.

Menurut Rosa dan Shalahuddin dalam Eva Meilinda (2016) mendifinisikan ”Sistem adalah kegiatan untuk melihat sistem yang sudah berjalan, melihat bagaimana yang bagus dan tidak bagus, dan kemudian mendokumentasikan kebutuhan yang akan dipenuhi dalam sistem yang baru.

Berdasarkan uraian diatas dapat disimpulkan bahwa sistem merupakan bagian yang saling berhubungan secara erat antara satu dengan yang lainnya. Sistem tersebut saling mempengaruhi untuk mencapai suatu tujuan yang sama.

2.1.2. Sistem Informasi

Sistem informasi adalah sekumpulan prosedur organisasi yang pada saat dilaksanakan akan memberikan informasi bagi pengambil keputusan atau mengendalikan organisasi (Hidayat, 2010:15).

Sedangkan menurut Pratama (2014:15) " Sistem informasi merupakan gabungan dari empat bagian utama keempat bagian utama tersebut mencakup perangkat lunak *(software),* perangkat keras *(hardware),* infrastruktur dan sumber daya manusia (SDM) yang terlatih".

Selain itu sistem informasi menurut Al-Fatta dalam (Sasongko, 2015) adalah sekumpulan objek-objek yang saling berelasi dan berinteraksi serta hubungan antar objek bisa dilihat sebagai satu kesatuan yang dirancang untuk mencapai satu tujuan.

Sedangkan menurut Tantra dalam (Maulana & Purwaningtias, 2016) sistem informasi adalah cara yang terorganisir untuk mengumpulkan, memasukkan dan memproses data dan menyimpannya, mengelola mengontrol dan melaporkannya sehingga dapat mendukung perusahaan, atau organisasi untuk mencapai tujuan.

Dari pendapat diatas dapat disimpulkan sistem informasi adalah sekumpulan prosedur organisasi yang saling berinteraksi dan merupakan gabungan empat bagian utama yaitu perangkat lunak *(software),* perangkat keras *(hardware),* infrastruktur dan sumber daya manusia (SDM) yang terlatih yang pada saat dilaksanakan akan memberikan informasi bagi pengambil keputusan atau mengendalikan informasi untuk mencapai suatu tujuan.

2.1.3. Karakteristik Sistem

Menurut Tohari (2014:2) suatu sistem mempunyai beberapa karakteristik, diantaranya yaitu sebagai berikut :

a) Komponen Sistem *(Component)*

Suatu sistem terdiri dari komponen-komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan.

b) Batasan Sistem *(Boundary)*

Batas sistem merupakan daerah yang membatasi antara sistem satu dengan sistem yang lainnya atau dengan lingkungan luarnya. Adanya batas sistem, maka sistem dapat membentuk suatu kesatuan, karena adanya batas sistem ini, fungsi dan tugas subsistem satu dengan yang lainnya berbeda tetapi tetap saling berinteraksi. Dengan kata lain, batas sistem merupakan ruang lingkup atau scope dari sistem atau subsistem itu sendiri.

c) Lingkungan Luar Sistem *(Environment)*

Lingkungan luar sistem adalah segala sesuatu diluar batas sistem yang mempengaruhi operasi suatu sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan merugikan. Lingkungan luar sistem yang bersifat menguntunbgkan harus dipelihara dan dijaga supaya tidak hilang pengaruhnya. Sedangkan, lingkungan luar yang bersifat merugikan harus dihilangkan supaya tidak mengganggu operasi dari sistem.

d) Penghubung Sistem *(Interface)*

Penghubung sistem merupakan suatu media (penghubung) antara satu subsistem dengan subsistem lainnya yang membentuk satu kesatuan, sehingga sumber-sumber daya mengalir dari subsistem yang satu ke

subsistem lainnya. Dengan kata lain, melalui penghubung, output dari subsistem akan menjadi input bagi subsistem lainnya.

e) Masukan Sistem *(Input)*

Input adalah energi atau sesuatu yang dimasukan kedalam suatu sistem yang dapat berupa masukan yaitu energi yang dimasukan supaya sistem dapat beroperasi atau masukan sinyal yang merupakan energi yang diproses untuk menghasilkan suatu luaran.

f) Luaran *(Output)*

Merupakan hasil dari energi yang diolah dan diklarifikasikan menjadi luaran yang berguna, juga merupakan luaran atau tujuan akhir dari sistem.

g) Pengolahan *(Process)*

Suatu sistem mempunyai bagian pengolah yang akan mengubah input menjadi output.

h) Sasaran *(Objective)*

Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuan.

2.1.4. Klasifikasi Sistem

Menurut Hutahaean (2015:6) Sistem dapat diklasifikasikan dalam beberapa sudut pandang antaranya adalah sebagai berikut :

1. Klasifikasi sistem sebagai :
   a. Sistem abstrak *(abstract sistem)*

      Sistem abstrak adalah sistem yang berupa pemikiran-pemikiran atau ide-ide yang tidak tampak secara fisik.

b. Sistem fisik *(physical sistem)*

Sistem fisik adalah sistem yang ada secara fisik.

2. Sistem diklasifikasikan sebagai :

a. Sistem alamiyah *(natural sistem)*

Sistem alamiyah adalah sistem yang terjadi melalui proses alam, tidak dibuat oleh manusia. Misalnya sistem perputaran bumi.

b. Sistem buatan manusia *(human made sistem)*

Sistem buatan manusia adalah sistem yang dibuat oleh manusia yang melibatkan interaksi antara manusia dengan mesin *(human machine sistem).*

3. Sistem diklasifikasikan sebagai :

a. Sistem tertentu *(deterministicl sistem)*

Sistem tertentu adalah sistem yang beroperasi dengan tingkah laku yang sudah dapat diprediksi, sebagai keluaran sistem yang dapat diramalkan.

b. Sistem tak tentu *(probalistic sistem)*

Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur *probabilistic*.

4. Sistem diklasifikasikan sebagai :

a. Sistem tertutup *(close sistem)*

Sistem tertutup adalah sistem yang tidak terpengaruhi dan tidak berhubungan dengan lingkungan luar, sistem bekerja otomatis tanpa ada turut campur lingkungan luar. Secara teoritis sistem tertutup ini ada, kenyataannya tidak ada sistem yang benar-benar tertutup, yang ada hanya *relatively closed sistem.*

b. Sistem terbuka *(open sistem)*

Sistem terbuka adalah sistem yang berhubungan dan terpengaruhi dengan lingkungan luarnya. Sistem ini menerima *input* dan *output* dari lingkungan luar atau subsistem lainnya. Karena sistem terbuka terpengaruh lingkungan luar maka harus mempunyai pengendalian yang baik.

2.1.5. Perancangan Sistem

Menurut Sofyan, Gustomi, & Fitrianto (2016) Perancangan atau desain didefinikan sebagai proses aplikasi berbagai teknik dan prinsip bagi tujuan pendefinisian suatu perangkat, suatu proses atau sistem dalam detail yang memadai untuk memungkinkan realisasi fisiknya.

Sedangkan menurut Mulyadi dalam (Sofyan, Gustomi, & Fitrianto, 2016) "Perancangan sistem merupakan penerjemahan kebutuhan pemakai informasi kedalam alternatif rancangan sistem informasi diajukan kepada pemakai informasi untuk dipertimbangkan".

2.1.6. Program

1. Pengertian program

Menurut Raharjo dalam (Yulia, 2017) program adalah "perangkat lunak *(software)* yang sebenarnya merupakan tuntunan instruksi yang ditulis dalam bentuk kode–kode menggunakan bahasa pemrograman tertentu dan telah dikompilasi dengan menggunakan *compiler* yang sesuai".

Sedangkan menurut Kadir dalam (Fadallah & Rosyida (2018) “Program adalah kumpulan instruksi yang digunakan untuk mengatur komputer agar melakukan suatu tindakan tertentu”.

Jadi program adalah kumpulan instruksi yang ditulis dalam bentuk kode-kode menggunakan bahasa pemrograman tertentu yang digunakan untuk mengatur komputer dengan maksud untuk melakukan suatu tindakan tertentu.

2. Pemrograman berorientasikan objek *(OOP)*

Menurut Rosa dan Shalahudin dalam (Mandiri & Octasia, 2016) “metodologi berorientasi objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasikan perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang diberlakukan terhadapnya”.

Pendapat lainnya “Pemrograman berorientasi objek adalah suatu cara baru dalam berpikir serta berlogika untuk menghadapi masalah-masalah yang akan dicoba atasi dengan bantuan komputer” (Fadallah & Rosyida, 2018).

Jadi pemrograman berorientasikan objek adalah suatu strategi pembangunan perangkat lunak yang mengorganisasi perangkat lunak sebagai kumpulan objek yang berisi data dan operasi yang mana membutuhkan cara berpikir atau logika yang dapat terselesaikan dengan bantuan komputer.

3. Visual Basic .NET

Microsoft Visual Basic.NET (VB.NET) adalah suatu pengembangan aplikasi bahasa pemrograman berbasis Visual Basic dan merupakan bahasa pemrograman terbaru buatan Microsoft setelah Microsoft Visual Basic 6.0. Pengembangan yang signifikan dari VB.NET ialah kemampuannya

memanfaatkan platform NET, sehingga pengguna dapat membuat aplikasi Windows, aplikasi konsol, pustaka kelas, layanan NT, aplikasi web form, dan XML Web Service, yang secara keseluruhan memungkinkan integrasi tanpa batas dengan bahasa pemrograman lain sehingga berpeluang untuk berintegrasi dengan web.

Menurut Didik Dwi Prasetyo dalam (Santoso, 2017) dalam bukunya 'Pemrograman Aplikasi *Database* dengan *Visual Basic .Net* 2005 dan *MS Access'* menuliskan bahwa :*Visual Basic .Net* merupakan salah satu bahasa pemrograman yang bisa digunakan untuk membangun aplikasi-aplikasi *.Net* di *platform Microsoft .Net.*

Menurut Nugraha, Syarif & Dharmawan (2018) *Visual Basic.NET* adalah generasiselanjutnya dari *Visual Basic. Visual Basic.NET* merupakan salah satu bahasa pemrograman yang terdapat pada paket program aplikasi *Visual Studio.NET* yangmenyediakan *tools* bagi para pengembang untuk membangun aplikasi yang berjalan di *.NET Framework.*

2.1.7. Basis data

Menurut Rosa dan Shalahuddin (2015:43) "basis data adalah media untuk menyimpan data agar dapat diakses dengan mudah dan cepat".

Sedangkan menurut Anhar dalam Yulia (2017) dalam "*Database* adalah sekumpulan tabel-tabel yang berisi data dan merupakan kumpulan dari *field* atau kolom. Struktur *file* yang menyusun sebuah *database* adalah Data *Record* dan *Field*".

Jadi basis data adalah media untuk menyimpan data yang mana merupakan tabel-tabel yang berisi data dan merupakan kumpulan dari *field* dan kolom.

A. PhpMyAdmin

PhpMyadmin adalah sebuah *software* yang berbentuk seperti halaman situs yang terdapat pada *web server* (Isty & Afifah, 2018).

Sedangkan menurut Zaki dan Smitdev dalam (Kristania et al., 2017) "*PHPMyAdmin* adalah *MySQL client* yang berupa aplikasi *web* dan umumnya tersedia di *server php* seperti *XAMPP* maupun *server* komersial lainnya.".

Dapat disimpulkan bahwa *PHPMyAdmin* adalah sebuah *software* yang bentuknya seperti halaman situs pada *web server* dan dapat juga diakses seperti pada *server php* seperti *XAMPP*, *WAMP*, dan *server* komersial lainnya.

B. MySQL

Menurut Sutanto dalam (Purba, 2015), *My Structured Query Langguage (MySQL)* adalah sebuah perangkat lunak sistem manajemen basis data yang *multithread, multi-user,* dengan sekitar 6 juta instalasi diseluruh dunia. MySQL AB membuat MySQL teredia sebagai perangkat lunak gratis dibawah lisensi GNU GPL *(General Public License*).

Sedangkan menurut Kadir dalam (Taufik, 2017) "MySQL (baca: mai-se-kyu-el) merupakan *software* yang tergolong *database server* dan bersifat *Open Source*".

Menurut Faisal dalam (Risdiansyah, 2017) bahwa "*MySQL* merupakan *database server* yang bersifat multiuser dan multi-threaded. *SQL* adalah bahasa database standar yang memudahkan penyimpanan, pengubahan dan akses informasi. Pada *MySQL* dikenal istilah database dan tabel. Tabel adalah sebuah struktur data dua dimensi yang terdiri dari baris-baris record dan kolom.

Sementara itu menurut Sadeli dalam (Isty & Afifah, 2018) "*MySQL* adalah *database* yang menghubungkan *script php* menggunakan perintah *query* dan *escaps character* yang sama dengan PHP.

Dari pendapat-pendapat diatas dapat disimpulkan *MySQL* adalah *database* yang menghubungkan *script php* menggunakan perintah *query* dan *escaps character* yang sama dengan php dan bersifat *Open Source*.

Sedangkan ODBC Menurut (M. A. Lubis, 2014) *Open Database Connectivity* (ODBC) adalah strategi *interface Microsoft* untuk pengaksesan data dalam lingkungan yang heterogen, baik untuk DMBS relational maupun non-relational.

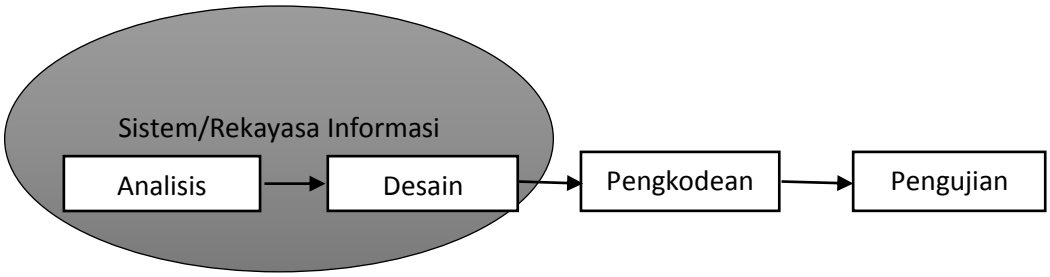
Jadi dapat disimpulkan bahwa *Open Database Connectivity* (ODBC) adalah untuk konektivitas untuk mengakses antar basis data.

2.1.8. Model pengembangan perangkat lunak

Metode yang digunakan pada pengembangan perangkat lunak ini menggunakan *waterfall* Rosa dan Shalahuddin (2015:28) yang terbagi dalam beberapa bagian, yaitu :

Gambar II.1

Ilustrasi Model Waterfall



Sumber : Rosa dan Shalahuddin (2015:28)

1. Analisis kebutuhan perangkat lunak

   Proses pengumpulan kebutuhan dilakukan secara intensif untuk mespesifikasikan kebutuhan perangkat lunak agar dapat dipahami perangkat lunak seperti apa yang dibutuhkan oleh *user* . Spesifikasi kebutuhan perangkat lunak pada tahap ini perlu untuk didokumentasikan.

2. Desain

   Desain perangkat lunak adalah proses multi langkah yang focus pada desain pembuatan program perangkat lunak termasuk struktur data, arsitektur perangkat lunak, representasi antar muka, dan prosedur pengkodean. Tahap ini mentranslasi kebutuhan perangkat lunak dari tahap analisis kebutuhan ke representasi desain agar dapat diimplementasikan menjadi program pada tahap selanjutnya. Desain perangkat lunak yang dihasilkan pada tahap ini juga perlu didokumentasikan.

3. Pembuatan kode program

   Desain harus ditranslasikan kedalam program perangkat lunak. Hasil dari tahap ini adalah program komputer sesuai dengan desain yang telah dibuat pada tahap desain.

4. Pengujian

   Pengujian focus pada perangkat lunak secara dari segi lojik dan fungsional dan memastikan bahwa semua bagian sudah diuji. Hal inni dilakukan untuk meminimalisir kesalahan *(error)* dan memastikan keluaran yang dihasilkan sesuai dengan keinginan.

5. Pendukung *(support)* atau pemeliharaan *(maintenance)*

Tidak menutup kemungkinan sebuah perangkat lunak mengalami perubahan ketika sudah dikirim ke user. Perubahan bisa terjadi karena adanya kesalahan yang muncul dan tidak terdeteksi saat pengujian atau perangkat lunak harus beradaptasi dengan lingkungan baru. Tahap pendukung atau pemeliharaan dapat mengulangi proses pengembangan mulai dari analisis spesifikasi untuk perubahan perangkat lunak yang sudah ada, tapi tidak untuk membuat perangkat lunak baru.

2.1.9. Persediaan Barang

Menurut Baridwan dalam (Palupi & Suprajang, 2015) menyatakan bahwa persediaan merupakan barang-barang yang dimiliki untuk dijual kembali atau digunakan untuk dijual kembali atau digunakan untuk memproduksi barang-barang yang digunakan untuk memproduksi barang-barang yang akan dijual.

Sedangkan Menurut Sofyan Assauri dalam (Jiang, Wang, & Wang, 2018) Persediaan barang ialah sebagai suatu aktiva lancar yang meliputi barang- barang yang merupakan milik perusahaan dengan maksud supaya dijual dalam suatu periode usaha normal ataupun persediaan barang-barang yang masih dalam pekerjaan sebuah proses menghemat ongkos penyimpanan dan pemeliharaan terhadap persediaan tersebut.

Dari berbagai definisi serta pengertian diatas, maka dapat diambil kesimpulan bahwa persediaan adalah merupakan barang-barang yang dimiliki untuk dijual kembali atau untuk digunakan memproduksi barang-barang yang akan dijual dan berpengaruh terhadap laporan keuangan perusahaan.

2.2. Teori Pendukung

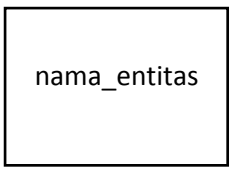
2.2.1. Entity Relationship Diagram (ERD)

Menurut Fatta dalam (Taufik, 2017) *Entity Relationship Diagram (ERD)* adalah gambar atau diagram yang menunjukan informasi dibuat, disimpan, dan digunakan dalam sistem bisnis.

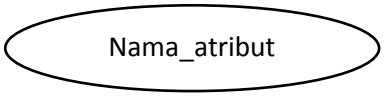
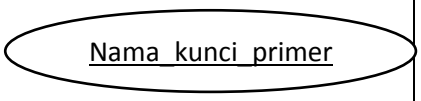
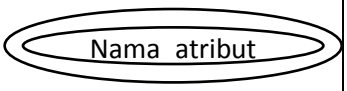
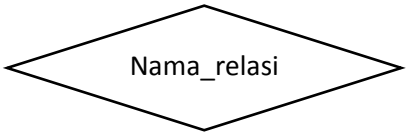
Sedangkan menurut Rosa dan Shalahuddin (2015:50) " ERD digunakan untuk pemodelan basis data relasional sehingga jika penyimpanan basis data menggunakan *OODBMS* maka perancangan basis data tidak perlu menggunakan ERD. ERD memiliki beberapa aliran notasi seperti notasi Chen (dikembangkan oleh Peter Chen), Barker (dikembangkan oleh Richard Barker, Ian palmer, Harry Ellis), notasi Crow's Foot, dan beberapa notasi lainnya. Namun yang banyak digunakan adalah notasi dari Chen, berikut adalah simbol-simbol yang digunakan ERD dengan notasi Chen :

Tabel II.1

Entity Relationship Diagram (ERD)

Simbol	Deskripsi
Entitas/*entity* 	Entitas merupakan data inti yang akan disimpan; bakal tabel pada basis data; benda yang memiliki data dan harus disimpan datanya agar dapat diakses oleh aplikasi komputer; penamaan entitas

Sumber : Rosa dan Shalahuddin (2015 : 50)

	biasanya lebih ke kata benda dan belum merupakan nama tabel
Atribut 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas
Atribut kunci primer 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas dan digunakan sebagai kunci akses *record* yang diinginkan; biasanya berupa id; kunci primer dapat lebih dari satu kolom, asalkan kombinasi dari beberapa kolom tersebut dapat bersifat unik (berbeda tanpa ada yang sama)
Atribut multinilai/*multivalue* 	*Field* atau kolom data yang butuh disimpan dalam suatu entitas yang dapat memiliki nilai lebih dari satu
Relasi 	Relasi yang menghubungkan antar entitas; biasanya diawali dengan kata kerja
Asosiasi / *association* 	Penghubung antara relasi dan entitas dimana kedua ujungnya memiliki *multiplicity* kemungkinan jumlah pemakaian .Kemungkinan jumlah maksimum keterhubungan antara entitas

Sumber : Rosa dan Shalahuddin (2015 : 50)

	satu dengan entitas yang lain disebut dengan kardinalitas 1 ke N atau sering disebut dengan *one to many* menghubungkan entitas A dan entitas B

Sumber : Rosa dan Shalahuddin (2015 : 50)

Jadi ERD *(Entity Relationship Diagram)* adalah gambar atau diagram yang menunjukan informasi dibuat, disimpan, dan digunakan dalam sistem bisnis. ERD ini digunakan untuk pemodelan basis data relasional.

2.2.2. Logical Record Structure (LRS)

Menurut Kusrini (2007:18) "*Logical Record Structure* merupakan representasi dari struktur *record- record* pada tabel-tabel yang terbentuk dari hasil relasi antar himpunan entitas pada diagram E-R."

Sedangkan menurut Friyadie dalam Taufik (2017) "sebelum tabel dibentuk dari *field* atau atribut entitas secara fisik atau level internal, maka harus dibuatkan suatu bentuk relational model yang dibuat secara *logic* atau *level external* dan konsep, dari pernyataan tersebut dibutuhkan yang disebut dengan *Logical Record Structure (LRS)".*

Berdasarkan pengertian menurut para ahli diatas dapat disimpulkan LRS *(Logical Record Structure)* dibutuhkan sebelum tabel dibentuk dari *field* atau atribut entitas secara fisik atau *level external* dan konsep. LRS sendiri merupakan representasi dari struktur *record-record* pada tabel-tabel yang terbentuk dari hasil kelas antar himpunan entitas pada diagram E-R.

2.2.3. Unified Modelling Language (UML)

Menurut Rosa Dan Shalahuddin (2015:133) "*UML (Unified Modelling Language)* adalah salah satu standar bahasa yang banyak digunakan didunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasikan objek.

Sedangkan menurut Ariani R. Sukamto dalam Taufik (2017) "UML merupakan bahasa *visual* untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram teks-teks pendukung".

Pendapat lainnya menurut Fowler, M. dalam (B. O. Lubis, 2016) *UML (Unified Modeling Language)* adalah "Keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientansi objek (OO). Definisi ini merupakan definisi yang sederhana".

Jadi *UML (Unified Modeling Language)* dapat diartikan sebagai bahasa visual untuk menggambarkan definisi-definisi tentang requirement, membuat analisis dan desain serta menggambar arsitektur dalam pemrograman berorientasikan objek dengan menggunakan teks-teks pendukung.

Menurut Rosa dan Shalahuddin (2015: 140) UML ini terdiri dari 13 macam diagram namun hanya beberapa diagram yang digunakan, diantaranya :

1. Activity Diagram

Diagram aktivitas atau *activity* diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak (Rosa dan Shalahuddin, 2015: 161).

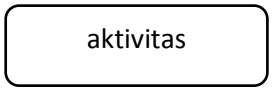
Menurut Rosa dan Shalahuddin (2015: 161) Diagram aktivitas juga banyak digunakan untuk mendefinisi hal-hal berikut :

- Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan
- Urutan atau pengelompokan tampilan dari sistem / *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antar muka tampilan
- Rancangan pengujian dimana setiap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kamus ujinya
- Rancangan menu yang ditampilkan pada perangkat lunak

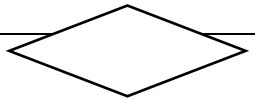
Berikut adalah simbol-simbol yang ada pada diagram aktivitas :

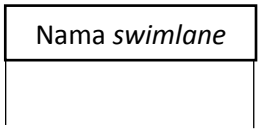
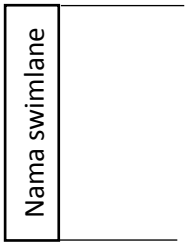
Tabel II.2

Activity Diagram

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas

Sumber : Rosa dan Shalahuddin (2015:162)

Percabangan / *decision* 	Asosiasi percabangan dimana jika

	ada pilihan aktivitas lebih dari satu
Penggabungan / *join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane  Atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Sumber : Rosa dan Shalahuddin (2015:162)

2. Use Case Diagram

Menurut Rosa dan Shalahuddin (2015: 155) *Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat.

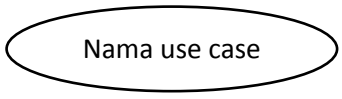
Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada use case yaitu pendefinisian apa yang disebut Aktor dan *use case*.

- 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.
- *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

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

Tabel II.3

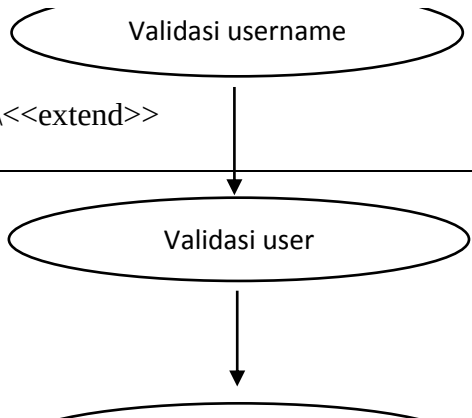
Use Case Diagram

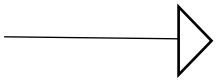
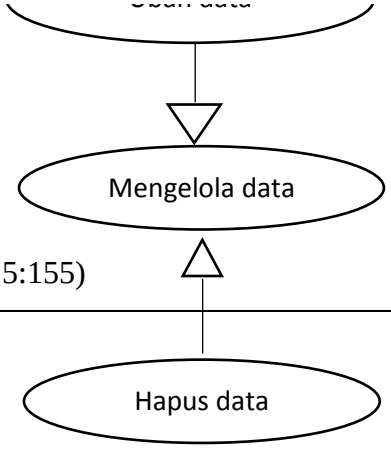
Simbol	Deskripsi
Use case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau Aktor; biasanya dinyatakan dengan menggunakan kata kerja diawal frase nama *use case*
 Nama actor	berinteraksi dengan sistem informasi yang akan dibuat diluar sistem

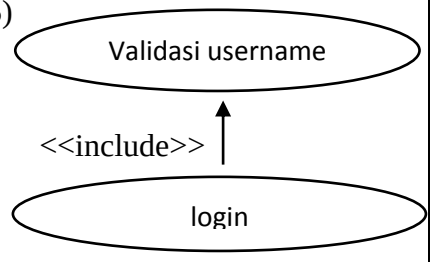
Sumber : Rosa dan Shalahuddin (2015:155)

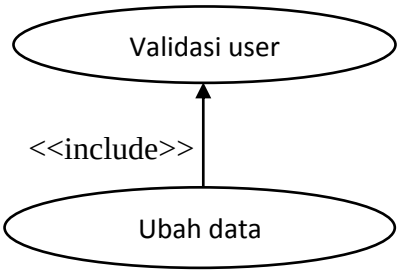
	informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari Aktor adalah gambar orang, tapi belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda diawal frase nama actor
Asosiasi / *association* _____	Komunikasi antara aktor dan *use case* yang berpartisipasi pada use case atau *use case* memiliki interaksi dengan actor
Ekstensi / *extend* <<extend>> ——→	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misalnya

Sumber : Rosa dan Shalahuddin (2015:155)



	<<extend>> 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
Generalisasi/ *generalization*  Sumber : Rosa dan Shalahuddin (2015:155)	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya :  Sumber : Rosa dan Shalahuddin (2015:155)

	Arah panah mengarah pada *use case* yang menjadi generalisasinya (umum)
Menggunakan / *include* / *uses* <<*include*>> <<*uses*>> Sumber : Rosa dan Shalahuddin (2015:155)	Relasi *use case* 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* : • *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, Misal pada kasus berikut :  • *Include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan,

	misal pada kasus :  Kedua interprestasi diatas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interprestasi yang dibutuhkan.

Sumber : Rosa dan Shalahuddin (2015:155)

3. Class Diagram

Menurut Rosa dan Shalahuddin (2015: 141) diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut pola dan metode atau operasi ;

- Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas
- Operasi atau metode adalah fungsi yang dimiliki oleh suatu kelas

Berikut adalah simbol-simbol yang ada pada diagram kelas :

Tabel II.4

Class Diagram

Simbol	Deskripsi
Kelas Nama_kelas + atribut - operasi	Kelas pada struktur system
Antarmuka / *interface*	Sama dengan konsep *interfac*e dalam

 Nama_interface	pemrograman berorientasi objek
Asosiasi / *association*	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
Asosiasi berarah / *directed association*	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antarkelas dengan makna generalisasi-speasialisasi (umum – khusus)
Kebergantungan / *dependency* 	Kebergantungan antar kelas
Agregasi / *aggregation*	Relasi antarkelas dengan makna semua-bagian *(whole – part)*

Sumber : Rosa dan Shalahuddin (2015 : 141)

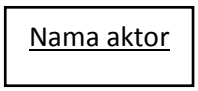
4. Sequence Diagram

Menurut Rosa dan Shalahuddin (2015: 165) diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek.

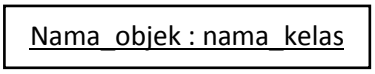
Berikut adalah simbol-simbol yang ada pada diagram sekuen :

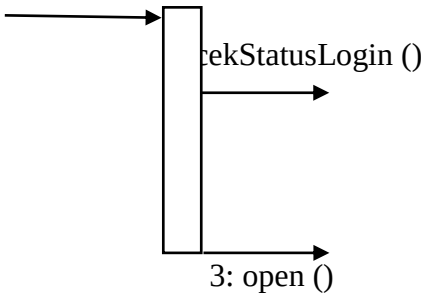
Tabel II.5

Sequence Diagram

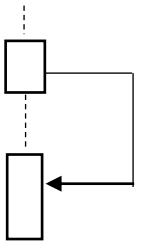
Symbol	Deskripsi
Aktor  Nama aktor atau 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari Aktor adalah gambar orang, tapi belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda

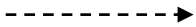
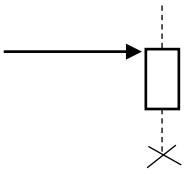
Sumber : Rosa dan Shalahuddin (2015:165)

Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek
Objek 	Menyatakan objek yang berinteraksi pesan
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya, misalnya

	1 : login ()  Maka cekStatusLogin () dan open () dilakukan di dalam metode login () Aktor tidak memiliki waktu aktif

Sumber : Rosa dan Shalahuddin (2015:165)

<<create>> 	objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe call 1 : nama_metode() 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri,  1 : nama_metode() 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
Pesan tipe *send* 1 : masukan	Menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek yang lainnya. Arah panah mengarah pada objek yang dikirimi
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
Pesan tipe *destroy* <<destroy>> 	Menyatakan suatu objek mengahiri hidup objek lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada *destroy*

Sumber : Rosa dan Shalahuddin (2015:165)

Sumber : Rosa dan Shalahuddin (2015:165)

2.2.4. XAMPP

Menurut Aryanto dalam (Kristania et al., 2017) “XAMPP merupakan sebuah aplikasi perangkat lunak pemrograman dan *database* yang di dalamnya terdapat berbagai macam aplikasi pemrograman seperti : *Apache HTTP server*, *MySQL, database*, Bahasa pemrograman *PHP* dan *perl*.”.

Menurut Riyanto dalam Isty & Afifah (2018) "XAMPP merupakan paket *web server* berbasis *open source* yang dapat dipasang pada beberapa sistem operasi yang ada *(Windows, Linux, dan Mac OS)"*.

Kesimpulannya XAMPP merupakan perangkat lunak pemrograman dan *database* atau paket web server berbasis *open source* yang dapat dipasang pada beberapa sistem operasi yang ada seperti *Windows*, *Linux*, dan *Mac OS*. Yang mana didalamnya terdapat berbagai macam aplikasi pemrograman seperti : Apache HTTP server, MySQL, *database*, Bahasa Pemrograman PHP dan Perl.

2.2.5. Crystal Report

Menurut Madcoms dalam Prasetyo (2017) *Crystal Report* merupakan program yang terpisah dengan program *Microsoft Visual Basic*, tetapi keduanya dapat dihubungkan *(linkage).*

Crystal Report merupakan piranti standart untuk pembuatan laporan pada sistem operasi windows, dimana cetakan / *template* laporan yang dihasilkan dapat disertakan pada banyak bahasa pemrogramman. (Prasetyo, 2017)

Dapat disimpulkan bahwa *Crystal Report* merupakan program atau piranti standar untuk pembuatan laporan pada sistem operasi *windows*, yang mana cetakan/template laporan yang dihasilkan dapat disertakan pada banyak bahasa pemrograman. *Crystal Report* ini terpisah dengan program *Microsoft Visual Basic*, tetapi keduanya dapat dihubungkan *(linkage).*