BAB II

TINJAUAN PUSTAKA

2.1. Teori Umum

2.1.1 Pengertian Komputer

Irma (2016:1), "Komputer adalah sekumpulan alat elektronik yang saling bekerja sama, dapat menerima data (*input*), mengolah data (proses) dan memberikan informasi (*output*) serta terkoordinasi dibawah kontrol program yang tersimpan dimemorinya."

Kadir (2017:2), "Komputer adalah peralatan elektronik yang bermanfaat untuk melaksanakan berbagai pekerjaan yang dilakukan oleh manusia."

Penulis menyimpulkan bahwa Komputer adalah alat bantu pemrosesan data secara elektronik mulai dari *input*, proses, dan *output* yang bermanfaat untuk melaksanakan berbagai pekerjaan manusia.

2.1.2 Pengertian Perangkat Lunak

Utami dan Asnawati (2015:2), "*Software* adalah prangkat lunak yang berisikan sebuah intruksi yang diperintahkan dan diproses dengan bantuan perangkat keras sehingga tanpa prangkat lunak, perangkat keras tidak bisa dipakai sehingga *software* dan *hardware* tidak bisa dipisahkan."

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*)."

Jadi didapatkan kesimpulan bahwa Perangkat Lunak adalah program komputer yang diperintahkan dan diproses dengan bantuan perangkat keras yang terasosiasi dengan dokumentasi perangkat lunak untuk memudahkan pekerjaan manusia.

2.1.3 Pengertian Basis Data

Sukamto dan Shalahuddin (2018:43), "Basis data adalah sistem komputerisasi yang tujuan utamanya adalah memelihara data yang sudah diolah atau informasi dan membuat informasi tersedia saat dibutuhkan."

Yanto (2016:11), "Basis data adalah kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redundansi), untuk memenuhi berbagai kebutuhan."

Kesimpulannya, Basis Data adalah sistem komputerisasi yang saling berhubungan berfungsi sebagai media penyimpanan data dan menyediakan informasi saat dibutuhkan.

2.1.4 Metode Pengembangan Sistem

Penelitian ini menggunakan metode pengembangan perangkat lunak dengan RUP (*Rational Unified Process*). Menurut Sukamto dan Shalahuddin (2018:125), "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 (2018:128-131) 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*).

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.

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 Judul

2.2.1 Pengertian Sistem

Pratama (2014:7), "Sistem didefinisikan sebagai sekumpulan prosedur yang saling berkaitan dan saling terhubung untuk melakukan suatu tugas bersama-sama."

Suryantara (2017:1), "Sistem terdiri atas komponen-komponen yang saling berhubungan satu sama lain dan bekerja sama untuk mencapai suatu tujuan."

Dari definisi diatas penulis menyimpulkan bahwa sistem adalah sekumpulan komponen yang saling berhubungan satu sama lain dan bekerja sama untuk mencapai suatu tujuan.

2.2.2 Pengertian Informasi

Anggraeni dan Rita (2017:2), menjelaskan bahwa "Informasi merupakan suatu kombinasi teratur dari orang-orang, *hardware*, *software*, jaringan komunikasi dan sumber daya data yang mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi."

Muslihudin dan Oktafianto (2016:9), menjelaskan bahwa "Informasi merupakan data yang diolah menjadi bentuk yang berguna untuk membuat keputusan."

Berdasarkan pengertian diatas penulis menyimpulkan bahwa informasi adalah data yang telah diolah melalui proses mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi untuk membuat keputusan.

2.2.3 Pengertian Sistem Informasi

Suryantara (2017:2), "Sistem informasi dapat dimaknai sebagai Suatu system yang dibuat oleh manusia yang terdiri atas komponen-komponen dalam organisasi untuk mencapai suatu tujuan, yaitu untuk menyajikan informasi."

Indrajani (2014:3), "Sistem Informasi merupakan kombinasi teratur apa pun dari orang-orang, hardware, software, jaringan komunikasi, dan sumber daya data, yang mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi."

Jadi dapat penulis simpulkan pengertian sistem informasi adalah kombinasi teratur apa pun dari orang-orang, *hardware*, *software*, jaringan komunikasi, dan sumber daya data untuk mencapai suatu tujuan yaitu untuk menyajikan informasi.

2.2.4 Pengertian Usaha Kecil dan Usaha Menengah

Definisi UMKM yang disetujui bersama DPR RI dan Presiden Republik Indonesia, yaitu: Dalam Pasal 1 Undang-Undang Nomor 20 Tahun 2008 tentang Usaha Mikro Kecil dan Menengah bahwa yang dimaksud dengan:

2.2.4.1 Pengertian Usaha Kecil

"Usaha Kecil adalah usaha ekonomi produktif yang berdiri sendiri, yang dilakukan oleh orang perorangan atau badan usaha yang bukan merupakan anak perusahaan atau bukan cabang perusahaan yang dimiliki, dikuasai atau menjadi bagian baik langsung maupun tidak langsung dari usaha menengah atau usaha

besar yang memenuhi kriteria usaha kecil sebagaimana dimaksud dalam Undang-Undang ini."

2.2.4.2 Pengertian Usaha Menengah

"Usaha Menengah adalah usaha ekonomi produktif yang berdiri sendiri, yang dilakukan oleh orang perseorangan atau badan usaha yang bukan merupakan anak perusahaan atau cabang perusahaan yang dimiliki, dikuasai atau menjadi bagian baik langsung maupun tidak langsung dengan usaha kecil atau besar dengan jumlah kekayaan bersih atau hasil penjualan tahunan sebagaimana diatur dalam undang-undang ini."

2.2.5 Pengertian Sistem Informasi Usaha Kecil dan Menengah (UKM) Kota Palembang.

"Sistem Informasi Usaha Kecil dan Menengah (UKM) Kota Palembang adalah Sistem Informasi yang dibuat untuk membantu Pelaku Usaha Kecil dan Menengah (UKM) dalam menjual dan mempromosikan produk unggulannya yang dapat diakses oleh masyarakat luas dari kota Palembang untuk meningkatkan ekonomi Pelaku UKM di Kota Palembang."

2.3 Teori Khusus

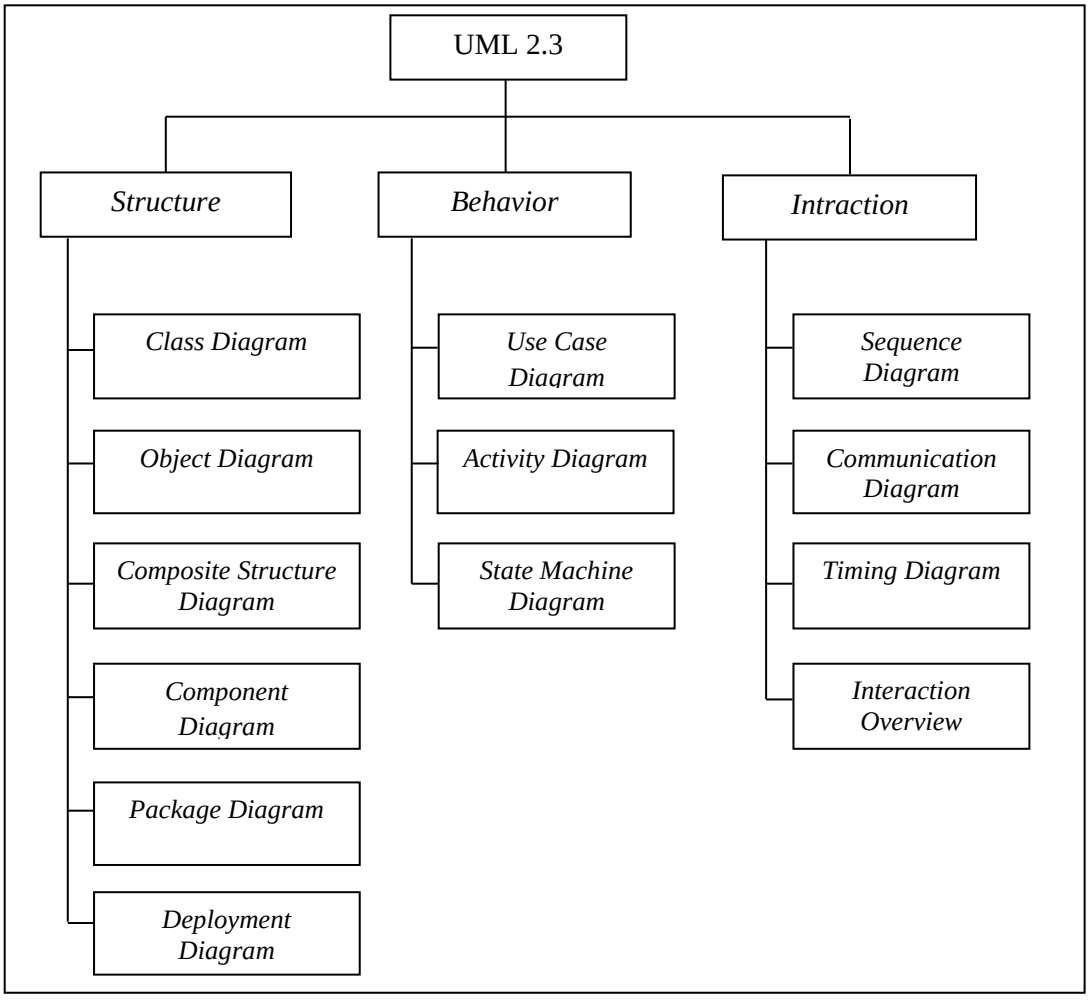
2.3.1 Pengertian *Unified Modeling Language* (UML)

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."

Gambar 2.1 Tampilan Logo 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

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.

2.3.2 Jenis-Jenis Diagram UML

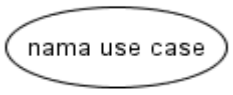
2.3.2.1 Pengertian *Use case* Diagram

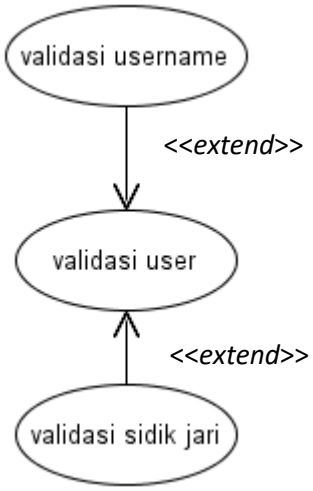
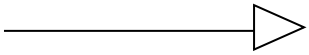
Sukamto dan Shalahuddin (2018:155), menjelaskan tentang *use case* diagram sebagai berikut :

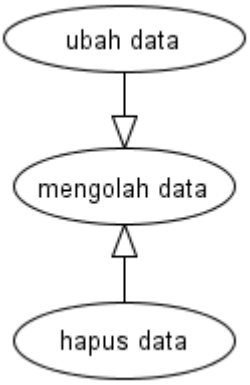
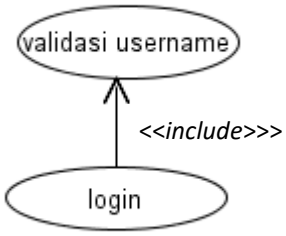
"Use case merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem."

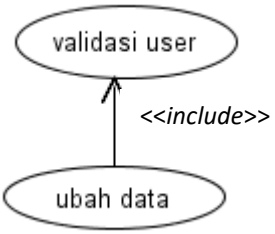
Adapun simbol-simbol yang digunakan dalam *Use case* adalah sebagai berikut:

Tabel 2.1. Simbol-simbol Use case Diagram

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 di awal-awal frase nama *use case*

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
asosiasi / *association* _____	komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki ingeraksi dengan actor
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  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* 	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*>> <<*uses*>>	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*: • *Include* berarti *use case* yang ditambahkan akan selalu di panggil saat *use case* tambahan dijalankan, misalnya pada kasus berikut:  • *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.

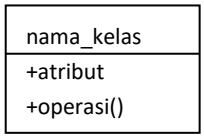
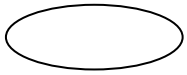
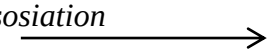
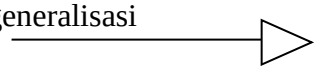
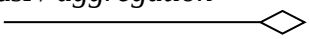
2.3.2.2 Pengertian *Class Diagram*

Sukamto dan Shalahuddin (2018:141), menjelaskan tentang *class diagram* sebagai berikut :

"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*

Simbol	Deskripsi
kelas 	Kelas pada struktur sistem
antarmuka / interface  nama_interface	Sama dengan konsep interface dalam pemrograman berorientasi objek
asosiasi / association	Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai *multiplicity*
asosiasi berarah / *directed assosiation* 	Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
generalisasi 	Relasi antarkelas dengan makna generalisasi – spesialisasi (umum khusus)
kebergantungan / *dependency* 	Relasi antarkelas dengan makna kebergantungan antar kelas
agregasi / *aggregation* 	Relasi antarkelas dengan makna semua-bagian (*whole-part*)

Sumber: Sukamto dan Shalahuddin (2018:146)

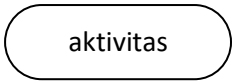
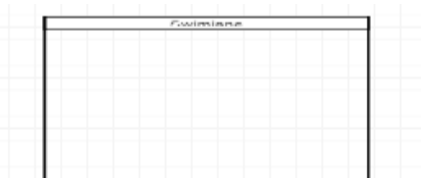
2.3.2.3 Pengertian *Activity Diagram*

Sukamto dan Shalahuddin (2018:161), menjelaskan tentang *activity diagram* sebagai berikut :

"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*

Simbol	Deskripsi
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* 	Asosiasi percabangan di mana 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: Sukamto dan Shalahuddin (2018:164)

2.3.2.4 Pengertian *Sequence Diagram*

(Sukamto dan Shalahuddin, 2018:165) “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

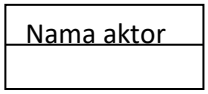
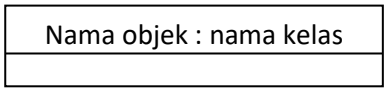
menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah beserta metode-metode yang dimiliki kelas yang diinstansikan menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat skenario yang ada pada *Use case*"

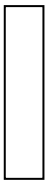
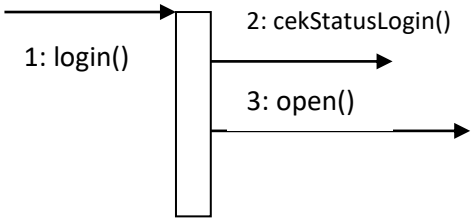
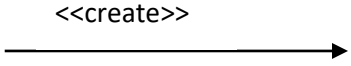
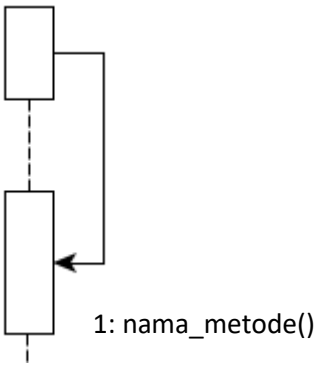
Banyaknya diagram sekuen yang harus digambarkan adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka diagram sekuen yang harus dibuat juga semakin banyak.

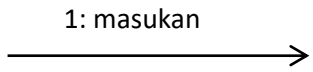
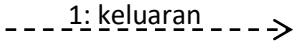
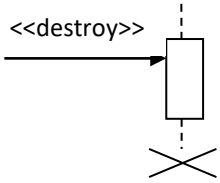
Dapat penulis simpulkan bahwa *Sequence diagram* adalah penggambaran skenario dari sebuah objek yang ada pada *use case* yang meliputi rangkaian langkah-langkah aktivitas dari objek berdasarkan waktu hidup objek dan pesan-pesan yang diterima maupun yang dikirimkan objek kepada objek lainnya.

Berikut simbol-simbol pada *Sequence Diagram* :

Tabel 2.4 Simbol-simbol pada *Sequence Diagram*

Simbol	Deskripsi
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
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 di dalamnya, misalnya  maka cekStatusLogin () dan open() dilakukan di dalam metode login() aktor tidak memiliki waktu aktif
Pesan tipe *create* 	menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
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

Pesan tipe *send* 	menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi
Pesan tipe *return* 	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 	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)

2.4 Teori Program

2.4.1 Pengertian XAMPP

Dadan dan Kerendi (2015:28), "*XAMPP* adalah salah satu aplikasi *web server apache* yang terintegrasi dengan *mysql* dan *phpmyadmin*."

Madcoms (2016:186), "*XAMPP* adalah sebuah paket kumpulan software yang terdiri dari *Apache, MySQL, PhpMyAdmin, Perl, Filezilla* dan lain-lain."

Jadi, penulis simpulkan pengertian *XAMPP* adalah sebuah aplikasi perangkat lunak pemrograman dan *database* yang di dalamnya terdapat berbagai macam aplikasi pemrograman yang terdiri dari Apache, *MySQL, PhpMyAdmin, Perl, Filezilla* dan lain-lain.

2.4.2 Pengertian *MySQL*

Sukamto dan Shalahuddin (2018:46), "SQL (*Structured Query Language*) adalah bahasa yang digunakan untuk mengelola data pada RDBMS. SQL awalnya dikembangkan berdasarkan teori aljabar relasional dan kalkulus."

Murya (2014:46), "MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (*database management system*) atau DBMS yang *multi thread, multi-user,* dengan sekitar 6 juta instalasi di seluruh dunia."

Jadi dapat penulis simpulkan pengertian MySQL adalah bahasa yang digunakan untuk mengelola data SQL (*database management system*) atau DBMS yang *multitheard, multi-user.*

2.4.3 Pengertian *PHP*

Murya (2014:65), "PHP *Hypertext Preprocessor* atau sering disebut PHP merupakan bahasa pemrograman berbasis *server-side* yang dapat melakukan parsing *script* php menjadi *script* web sehingga dari sisi *client* menghasilkan suatu tampilan yang menarik."

Madcoms (2016:2), "PHP (*Hypertext preprocessor)* adalah bahasa *script* yang dapat ditanamkan atau disisipkan ke dalam HTML."

Jadi, dapat penulis simpulkan pengertian PHP adalah bahasa pemograman berbasis *server-side* yang bisa kita gunakan untuk membuat aplikasi web yang disisipkan pada HTML.

Gambar 2.3. *Tampilan Logo PHP*

2.4.3.1 Sintaks Dasar PHP

Yuana (2015:2), menjelaskan kode-kode PHP dituliskan diantara tanda berikut ini:

```
<?php
  …
  …
```

…

?>

Atau

<?

…

…

…

?>

Apabila membuat kode PHP dan berencana akan mendistribusikan ke pihak/orang lain, maka usahakan menggunakan sintaks <?php … ?>. Hal ini dikarenakan untuk penggunaan kode yang menggunakan <? … ?> terkadang tidak bisa dijalankan dalam *server* tertentu.

2.4.3.2 Tipe Data PHP

Tipe data merupakan jenis dari suatu data yang akan diproses oleh bahasa pemrograman. Murya (2014:26), menjelaskan beberapa tipe data dalam PHP, sebagai berikut :

1. ***Integer*** merupakan tipe data yang berguna untuk menyimpan bilangan bulat. *Range* bilangan *integer* adalah antara -2.147.4833.647 sampai dengan 2.147.483.647
2. ***Double Floating*** adalah tipe data yang berguna untuk menyimpan bilangan desimal. *Range* bilangan floating point antara 1e308 sampai dengan 1e308.
3. ***Boolean*** adalah tipe data yang paling sederhana, hanya berupa ***TRUE*** dan ***FALSE***.
4. ***String*** adalah tipe data yang terdiri dari kata, bias berupa kata tunggal maupun kalimat. Penulisan *string* harus diapit dengan tanda petik, baik berupa petik tunggal (‘…’) maupun petik ganda (“...”).
5. ***Objek*** adalah tipe data dibuat dengan tujuan agar para *programmer* terbiasa dengan OOP. Tipe data ini bias berupa bilangan.
6. ***Array*** merupakan ***Tipe Compound Primitif,*** terdapat pada bahasa pemrograman lain.

7. ***Null*** adalah tipe data yang tidak memuat apapun. Setiap variabel yang diset menjadi tipe data Null, ini akan menjadikan variabel tersebut kosong.
8. ***Resources*** tipe data spesial yang satu ini dikhususkan untuk menyimpan *resources,* sumber atau alamat.

2.4.4 Pengertian PHPMyAdmin

Hikmah, dkk (2015:2), "PHPMyAdmin merupakan aplikasi yang dapat digunakan untuk membuat *database*, pengguna (*user*), memodifikasi tabel, maupun mengirim database secara cepat dan mudah tanpa harus menggunakan perintah (*command*) *SQL* ."

Rahman (2014:12), "PHPMyAdmin adalah aplikasi PHP sebagai administrator *MySQL,* PHPMyAdmin mendukung berbagai aktivitas *MySQL* seperti pengelolaan data, *table,* relasi antar *table* dan lain sebagainya."

Jadi, penulis simpulkan pengertian PHPMyAdmin adalah aplikasi PHP sebagai administrator *MySQL* yang digunakan untuk membuat *database,* mengelola tabel, mengelola data, relasi antar tabel, dan mengirim database secara praktis tanpa harus menggunakan perintah (*command*) *SQL*."

Gambar 2.4. Tampilan logo PHPMyAdmin