BAB II
TINJAUAN PUSTAKA

2.1 Teori Umum

2.1.1 Pengertian Komputer

Blissmer (Dikutip Susanto 2009:2), "Komputer adalah suatu alat elektronik yang mampu melakukan beberapa tugas seperti menerima input, memproses input tadi sesuai dengan programnya, menyimpan perintah-perintah dan hasil pengolahan, serta menyediakan output dalam bentuk informasi". Sedangkan, menurut Kadir (2017:2), "Komputer merupakan peralatan elektronik yang bermanfaat untuk melaksanakan berbagai pekerjaan yang dilakukan oleh manusia".

Dari kedua definisi diatas penulis menyimpulkan bahwa komputer merupakan perangkat keras yang dapat digunakan untuk membantu dalam berbagai pekerjaan seperti menerima input, memproses input hingga menghasilkan output dalam bentuk informasi. Komputer tentunya sangat bermanfaat bagi manusia, contohnya untuk pembuatan dokumen.

2.1.2 Pengertian Perangkat Lunak (*Software)*

Sutabri (2014:62), "Perangkat Keras komputer tidak berfungsi tanpa perangkat lunak. Teknologi yang canggih dari perangkat keras akan berfungsi bila intruksi-intruksi tertentu telah diberikan kepadanya. Intruksi-intruksi tersebut disebut dengan perangkat lunak. Perangkat Lunak dapat diklasifikasikan kedalam dua bagian besar, yaitu:

1. Perangkat Lunak Sistem (*system software*), yaitu perangkat lunak yang mengoperasikan sistem komputer. Perangkat lunak sistem dapat dikelompokkan menjadi 3 bagian, yaitu:
    a. Perangkat lunak sistem operasi (*operating system*), yaitu program yang ditulis untuk mengendalikan dan mengoordinasikan operasi dari sistem komputer.

b. Perangkat lunak sistem bantuan (*utility*), yaitu program yang ditulis untuk bantuan yang berhubungan dengan sistem komputer, misalnya memformat disc, menyalin disc, mencegah dan membersihkan virus.

c. Perangkat lunak bahasa (*laguage software*), yaitu program yang digunakan untuk menerjemahkan instruksi-instruksi yang ditulis dalam bahasa pemrograman kedalam bahasa mesin supaya dapat dimengerti oleh komputer.

2. Perangkat lunak aplikasi (*aplication software),* yaitu Program yang ditulis dan diterjemahkan oleh perangkat lunak bahasa untuk menyelasaikan suatu aplikasi tertentu". Begitupula menurut Kadir (2017:2), "Perangkat lunak adalah instruksi-instruksi yang ditujukan kepada komputer agar dapat melaksanakan tugas sesuai kehendak pemakai".

Dari beberapa definisi di atas penulis menyimpulkan bahwa perangkat lunak dapat diklasifikasi menjadi 2 bagian yaitu perangkat lunak sistem yaitu perangkat lunak yang mengoperasikan sistem komputer dan perangkat lunak aplikasi yaitu Program yang ditulis dan diterjemahkan oleh perangkat lunak bahasa untuk menyelasaikan suatu aplikasi tertentu, dan juga perangkat lunak digunakan untuk melaksanakan tugas sesuai kehendak pemakai.

2.1.3 Pengertian Perangkat Keras (*Hardware*)

Sutabri (2014:32), "Perangkat keras sebagai subsistem dari sistem komputer juga mempunyai komponen, yaitu

1. Komponen peranti masukkan (*input device*) yaitu alat yang digunakan untuk menerima masukkan data atau masukkan program misalnya *keyboard, scanner,* sensor, dan pengenal suara.
2. Komponen peranti pemeroses (*processing device*), yaitu alat di mana instruksi-instruksi program dieksekusi untuk memproses data yang dimasukkan lewat alat masukkan yang hasilnya nanti akan ditampilkan dialat output misalnya *CPU (central processing unit).*
3. Komponen peranti keluaran (*output device),* yaitu alat untuk menampilkan pengolahan data seperti tulisan (huruf, kata, angka, karakter, dan simbol-

simbol), *image* (bentuk grafik atau gambar) dan suara (bentuk musik) misalnya *hardcopy device (printer)* dan *softcopy device (monitor).*

4. Komponen peranti penyimpanan (*storage*) eksternal adalah penyimpanan luar, karena terletak diluar alat prosesnya atau disebut *mass storage* atau penyimpanan masal karena penyimpanan umumnya lebih besar dari *main memory".* Lain halnya menurut Kadir (2017:2), "Perangkat keras adalah peranti-peranti yang terkait dengan komputer dan terlihat secara fisik. Monitor, *hard disk, dan mouse* adalah contoh perangkat keras".

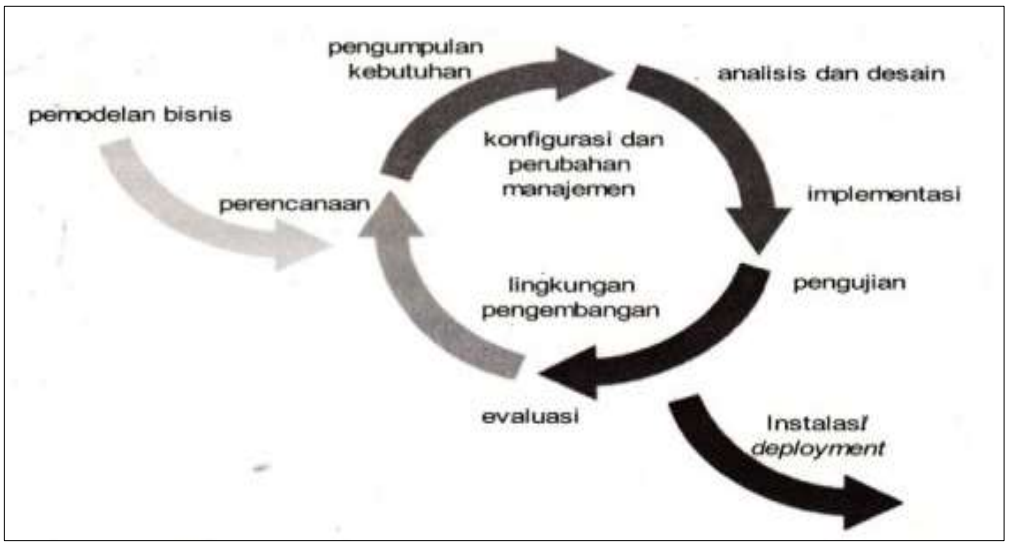
Dari beberapa definisi perangkat keras penulis menyimpulkan bahwa perangkat keras adalah beberapa komponen yang bekerja saling mendukung sesuai dengan instruksi *software.*

2.1.4 Metode Pengembangan Sistem

Menurut Sukamto dan Shalahuddin (dikutip Fitria dan Widiowati 2017:28), RUP (*Rational Unified Process*) merupakan pendekatan pengembangan perangkat lunak yang dilakukan berulangulang (*iterative*), fokus pada arsitektur (*architecture-centric*), lebih diarahkan berdasarkan penggunaan kasus (*use case driven*). RUP merupakan proses rekayasa perangkat lunak dengan pendefinisian yang baik (*well defined*) dan penstrukturan yang baik (*well structured*)".

IBM, 1998 (dalam Hutahean dkk 2019: 5790), "*Rational Unified Process* (RUP) merupakan salah satu pengembangan perangkat lunak dengan menggunakan pendekatan yang disiplin dalam melakukan setiap tugas dan tanggung jawabnya dalam sebuah organisasi. Tujuan dari RUP itu sendiri adalah untuk dapat menjamin produksi berkualitas tinggi dan memenuhi semua kebutuhan pihak yang berkepentingan, termasuk waktu dan biaya sesuai dengan rencana yang telah disepakati sebelumnya."

Proses pengulangan/iteratif pada RUP secara global dapat dilihat pada gambar berikut:



(*Sumber*: Sukamto dan Shalahuddin (dikutip Fitria dan Widiowati, 2017:29))

Gambar 2.1 Proses iteratif *RUP (Rational Unified Process)*

Menurut Sukamto dan Shalahuddin (2016:128-131), RUP memiliki empat buah tahap atau fase yang dapat dilakukan pula secara iteratif.berikut merupakan penjelasam untuk setiap fase pada RUP.

a. *Fase Inception* (Permulaan)

Tahap dimana kita memodelkan proses bisnis yang dibutuhkan (*business modeling*) dan mendefinisikan kebutuhan akan sistem yang akan dibuat (*requirements*). Berikut adalah tahap yang dibutuhkan pada tahap ini:

1. Memahami ruang lingkup dari proyek (termasuk pada biaya, waktu, kebutuhan, resiko dan lain sebagainya)
2. Membangun kasus bisnis yang dibutuhkan

Hasil yang diharapkan pada tahap ini adalah memenuhi *Lifecycle Ovjective Milestone* (batas/tonggak objektif dari siklus) dengan kriteria berikut:

1. Umpan balik dari pendefinisian ruang lingkup, perkiraan biaya, dan perkiraan jadwal

2. Kebutuhan dimengerti dengan pasti (dapat dibuktikan) dan sejalan dengan kasus primer yang dibutuhkan.
3. Kredibilitas dari perkiraan biaya, perkiraan jadwal, penentuan skala prioritas, resiko dan proses pengembangan.
4. Ruang lingkup purwarupa (*prototype*) yang akan dikembangkan
5. Membangun garis besar dengan membandingkan perencanaan yang direncanakan.

Pada fase ini, penulis menyimpulkan beberapa kegiatan yang dilakukan berdasarkan hal-hal diatas, diantaranya penyelidikan awal, studi kelayakan hingga kebutuhan fungsional dan non fungsional.

1. Penyelidikan Awal, pada hakikatnya adalah mencari jawaban atau menyelesaikan masalah dengan jawaban yang benar atau sekurang-kurangnya dapat mengetahui kebenaan yang logis melalui pemikiran yang mempunyai kesinambungan dengan fakta (Yahaya, dkk. 2007:2-3). Penulis menyimpulkan bahwa penyelidikan awal adalah suatu cara menguraikan masalah yang akan dihadapi untuk mencari jawaban dari permasalahan yang dihadapi.
2. Studi Kelayakan sendiri merupakan suatu usaha yang akan dilakukan dalam rangka menentukan apakah sebuah kebutuhan layak atau tidak nya untuk dijalankan. Terdapat beberapa faktor atau metode pada Studi Kelayakan dimana faktor pada Studi Kelayakan tidak ada yang baku, tingkat kerumitan, kedalaman dan kompleksitas studi kelayakan bergantung pada objek studi itu sendiri (Kasmir dan Jakfar, 2013:2-4).

Selanjutnya, penulis menentukan 3 studi kelayakan, diantaranya kelayakan teknis, operasional dan ekonomis, menurut Alkautsar dan Welda (2013:4), faktor kelayakan teknis berhubungan dengan *computer-oriented*, faktor kelayakan operasional berhubungan dengan *people-oriented*, dan faktor kelayakan ekonomis berhubungan dengan biaya dan keuntungan sistem informasi.

3. Kebutuhan fungsional menurut Setiyani dan Gintings (2019:407) adalah kebutuhan yang menggambarkan fungsionalitas sistem yang akan

dikembangkan atau di bangun. Fungsionalitas sistem ini merupakan harapan solusi yang diinginkan oleh pemangku kepentingan. Selanjutnya, menurut Gomez Sotelo et al. (dikutip Setiyani dan Gintings, 2019:407) kebutuhan non fungsional merupakan spesifikasi produk yang diharapkan dalam hal menangkap properti yang digunakan untuk mengoperasikan sistem atau properti perilaku yang dimiliki oleh sistem.

b. *Fase Elaboration* (Perencanaan)

Tahap ini lebih difokuskan pada perencanaan arsitektur sistem. Tahap ini lebih pada analisis dan desain sistem. Hasil yang diharapkan dari tahap ini memenuhi *Lifecycl Ovjective Milestone* (batas/tonggak objektif dari siklus) dengan kriteria, model kasus yang digunakan (*use case)* dimana kasus dan aktor yang terlibat telah diidentifikasikan dan sebagaian besar harus dikembangkan.

c. *Fase Construction (*Konstruksi)

Tahap dimana kita mengembangkan komponen dan fitur-fitur sistem. Implementasi dan pengujian sistem yang fokus pada implementasi perangkat lunak pada kode program. Hasil dari fase ini yaitu implementasi sistem informasi monitoring berbasis website dan hasil pengujian sistem yaitu dengan blackbox validation testing dan compatibility testing. Menurut Hariyanto dkk. (2018:270), terdapat dua teknik pengujian black-box yaitu validation testing dan compability testing.

Menurut Syed (dalam Jaya, 2018:45), Black-Box Testing merupakan Teknik pengujian perangkat lunak yang berfokus pada spesifikasi fungsional dari perangkat lunak. Black-Box Testing juga menurut Watkins (dalam Jaya, 2018:45-46), bekerja dengan mengabaikan struktur kontrol sehingga perhatiannya difokuskan pada informasi domain Blackbox Testing memungkinkan pengembang software untuk membuat himpunan kondisi input yang akan melatih seluruh syarat-syarat fungsional suatu program.

Menurut Amman (dalam Jaya, 2018:46) keuntungan penggunaan metode Blackbox Tetsting adalah:

a. Penguji tidak perlu memiliki pengetahuan tentang bahasa pemrograman tertentu;
b. Pengujian dilakukan dari sudut pandang pengguna, ini membantu untuk mengungkapkan ambiguitas atau inkonsistensi dalam spesifikasi persyaratan;
c. Programmer dan tester keduanya saling bergantung satu sama lain.

Sedangkan, menurut Amman (dalam Jaya, 2018:46) kekurangan dari metode Blackbox Testing adalah:

a. Uji kasus sulit disain tanpa spesifikasi yang jelas;
b. Kemungkinan memiliki pengulangan tes yang sudah dilakukan oleh programmer;
c. Beberapa bagian back end tidak diuji sama sekali.

d. Fase Transition (Transisi)

Tahap dimana kita deployment atau Instalasi sistem agar dapat dimengerti oleh user. Aktifitas pada tahap ini termasuk pada pelatihan user dan pemeliharaan. Produk perangkat lunak juga disesuaikan dengan kebutuhan yang didefinisikan pada tahap *inception*. Jika semua kriteria objektif maka dianggap sudah memenuhi *Product Release Milestone* (batas/tonggak peluncuran produk) dan pengembangan perangkat lunak selesai dilakukan.

Akhir dari keempat fase ini adalah produk perangkat lunak yang sudah lengkap, keempat fase pada RUP dijalankan secara urut dan berulang, dengan setiap iterasi digunakan untuk memperbaiki iterasi berikutnya.

2.2 Teori Khusus

2.2.1 *Unified Modelling Language* (*UML*)

Menurut Widodo (dikutip Pranata dkk. 2015:26), "*UML* adalah sebuah bahasa untuk menentukan, visualisasi, konstruksi, dan mendokumentasikan *artifact* (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat lunak). *Artifact* dapat berupa model, deskripsi atau perangkat lunak dari sistem perangkat lunak, seperti pada pemodelan bisnis dan sistem non perangkat lunak lainnya".

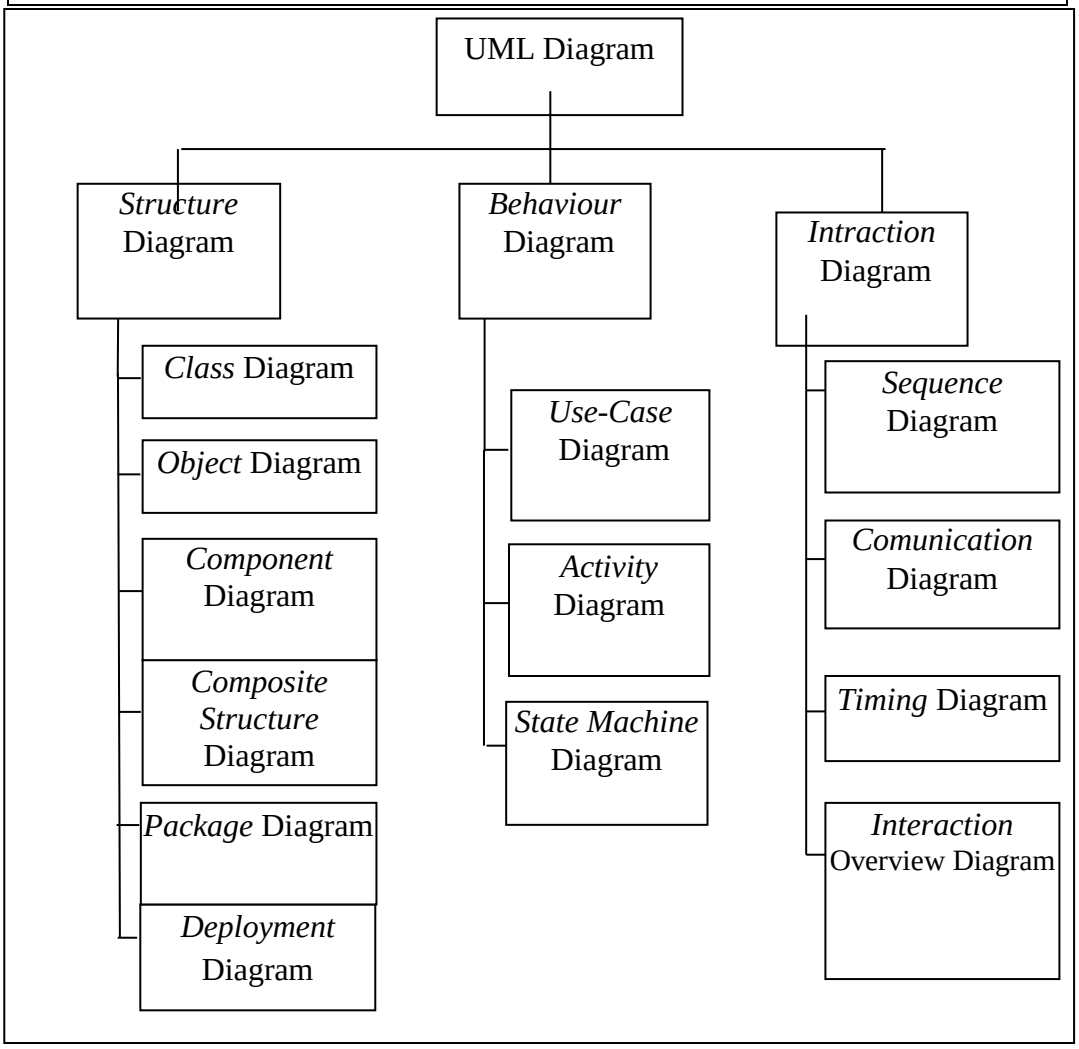
Adapun pendapat lain menurut Sukamto dan Shalahuddin (2016:133), "UML (*Unifed Modeling Language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek."

Dari pengertian diatas dapat disimpulkan bahwa *UML* (*Unified Modelling Language*) adalah sebuah bahasa yang berdasarkan grafik/gambar untuk memvisualisasikan hingga mendokumentasikan sebuah sistem pengembangan perangkat lunak berorientasi objek.

2.2.2 Klasifikasi Diagram *UML* (*Unified Modelling Language*)

Sukamto dan Shalahuddin (2016:140), "*UML* (*Unified Modelling Language*) terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori". Berikut ini penjelasan singkat dari pembagian kategori tersebut.

Struktur Klasifikasi Diagram UML (Unified Modelling Language)



(*Sumber*: Sukamto dan Shalahuddin (2016:140))

Gambar 2.2 Klasifikasi Diagram UML (*Unified Modelling Language*)

a. *Structure* Diagram

Yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

b. *Behavior* Diagram

Yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.

c. *Interaction* Diagram

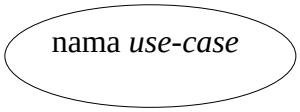
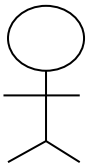
Yaitu kumpulan diagram yang digunakan untuk menggambarkan interkasi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

2.2.3 *Use-Case* Diagram

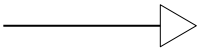
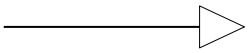
Menurut Widodo (dalam Pranata dkk. 2015:26), *Use-case* diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem, yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *Use-case* merepresentasikan sebuah interaksi antara aktor dengan sistem.

Adapun pendapat Sukamto dan Shalahuddin (2016:155), *"Use-case* merupakan pemodelan untuk kelakuan *(behavior)* sistem informasi yang akan dibuat. *Use-case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat". 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 frase nama *use-case*.
2.	Aktor / a*ctor*  nama aktor nama_*interface*	Orang, proses, atau sistem 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 berpartisipasi pada *use-case* atau *use-case* yang memiliki interaksi dengan actor.

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

4.	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; ditambahkan, missal 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.
6.	Menggunakan / *include* /*uses* <<*include*>>  <<*uses*>> 	Relasi *use-case* tambahan ke sebuah *use-case* di mana *use-case* yang ditambahkan memerlukan *use-case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use-case*.

(*Sumber*: Sukamto dan Shalahuddin (2016:156-158))

2.2.4 *Class* Diagram

Sukamto dan Shalahuddin (2016:141), 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 dan metode atau operasi.

a. Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas
b. Operasi atau model adalah fungsi-fungsi yang dimiliki oleh suatu kelas

Tabel 2.2 Simbol-simbol *Class* Diagram

No	Simbol	Deskripsi
1.	Kelas nama_kelas + attribut + operasi ()	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 dengan *multiplicity*.
4.	Asosiasi berarah / *directed association*	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/ d*ependency*	Relasi antarkelas dengan makna kebergantungan antarkelas.
7.	Agregasi / *aggregation*	Relasi antarkelas dengan makna semua-bagian *(whole-part).*

(*Sumber*: Sukamto dan Shalahuddin (2016:144-147)

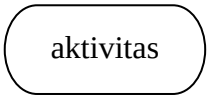
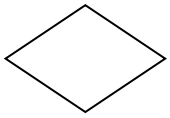
2.2.5 *Activity Diagram*

Sukamto dan Shalahuddin (2016:161), *Activity Diagram* atau Diagram Aktivitas 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.

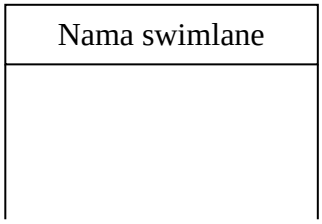
Diagram aktivitas juga banyak digunakan untuk mendefinisikan hal-hal berikut:

1. Rancangan proses bisnis dimana setiap urutan aktivitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan
2. Urutan atau pengelompokan tampilan dari sistem / *user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antarmuka tampilan
3. Rancangan pengujian dimana seriap aktivitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujinya
4. Rancangan menu yang ditampilkan pada perangkat lunak

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 dimana jika ada pilihan aktivitas lebih dari satu.

Lanjutan Tabel 2.3 Simbol-simbol *Activity* Diagram

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* 	*Swimlane* memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

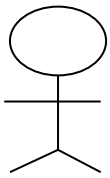
(*Sumber*: Sukamto dan Shalahuddin (2016:162-163))

2.2.6 *Sequence Diagram*

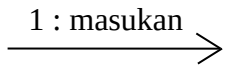
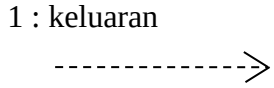
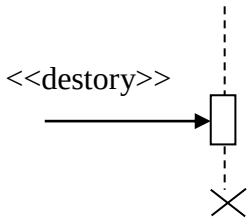
Sukamto dan Shalahuddin (2016:165), "Diagram sekuen atau *Sequence Diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karna itu, untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diintansiasi menjadi objek itu. Membuat diagram sekuen juga dibutuhkan untuk melihat sekenario yang ada pada *use case*.

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

Tabel 2.4 Simbol-simbol *Sequence Diagram*

No	Simbol	Deskripsi
1.	Aktor  Aktor Atau Nama_aktor tanpa waktu aktif	Orang, proses, atau sistem 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*.
2.	Garis hidup/ *lifeline* 	Menyatakan kehidupan suatu objek.
3.	Objek nama objek : nama kelas	Menyatakan objek yang berinteraksi pesan.
4.	Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya.
5.	pesan tipe *create* <<create>> 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat.
6.	Pesan tipe *call* 1 : nama_metode() 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri.

Lanjutan Tabel 2.4 Simbol-simbol *Sequence* Diagram

7.	Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data/ masukan/ informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
8.	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.
9.	Pesan tipe *destrory* 	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 (2016:165-167))

2.3 Teori Judul

2.3.1 Pengertian Aplikasi

Juansyah (2015:2), dalam kamus Komputer eksekutif, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu tehnik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang diinginkan atau diharapkan maupun pemrosesan data yang di harapkan.

Koyuko dkk (dalam Oki 2016), Aplikasi adalah suatu *software* yang dapat melakukan beberapa tugas tertentu seperti membuat dokumen, penambahan data dan beberapa tugas lainnya. *suite* aplikasi (*application suite*) adalah penggambungan dari berbagai macam aplikasi menjadi satu. salah satu contohnya *Microsoft Office* dan *Open Office* yaitu menggabungkan aplikasi *Microsoft word*,

Microsoft excel, dan beberapa aplikasi lainnya. Pada umumnya aplikasi-aplikasi yang berada dalam suatu paket mempunyai *interface* yang sama sehingga memudahkan *user* untuk mempelajari dan memakai aplikasi tersebut dan juga aplikasi- aplikasi tersebut mempunyai bisa berinteraksi antar aplikasi sehingga memudahkan *user*. Contohnya, suatu lembar kerja dapat disimpan dalam suatu dokumen pengolah kata meskipun dibuat pada aplikasi lembar kerja yang terpisah.

Dari definisi diatas penulis menyimpulkan bahwa aplikasi adalah perangkat lunak atau program komputer yang beroperasi pada sistem tertentu yang diciptakan dan dikembangkan untuk melakukan perintah tertentu.

2.3.2 Pengertian Pendaftaran

Putra (2019:88), Pendaftaran pada dasarnya berfungsi untuk pendataan dan pencatatan data seseorang, sehingga dapat terorganisir, teratur dengan cepat dan tepat dengan beberapa persyaratan yang telah ditentukan sebelumnya.

Adapun pendapat lain Menurut (DEPDIKBUD01), "Pendaftaran adalah proses, cara, perbuatan mendaftar yaitu pencatatan nama, alamat dsb dalam daftar".

Berdasarkan pernyataan yang dikemukakan oleh pengarang di atas, maka penulis mengambil kesimpulan bahwa pendaftaran adalah proses pencatatan identitas pendaftar sesuai dengan kebutuhan dan persyaratan yang dibutuhkan kedalam sebuah media penyimpanan yang digunakan dalam proses pendaftaran untuk suatu tujuan pendaftaran tertentu.

2.3.3 Pengertian Investor

Tandelilin (2010:3), Investor adalah pihak-pihak yang melakukan kegiatan investasi. Investor pada umumnya bisa digolongkan menjadi dua, yaitu investor individual (*individual/retail investors*) dan investor institusional (*institutional investors*). Investor individual terdiri dari individu-individu yang melakukan aktivitas investasi. Misalkan, Investor menginvestasikan dananya dalam bentuk saham akan disebut sebagai investor individual. Sedangkan investor institusional biasanya terdiri dari perusahaan-perusahaan asuransi, lembaga penyimpan dana

(bank dan lembaga simpan-pinjam), lembaga dana pensiun maupun perusahaan investasi.

2.3.4 Pengertian *Web*

Abdulloh (2018:1), *Website* dapat diartikan sebagai kumpulan halaman yang berisi informasi data digital baik berupa teks, gambar, animasi, suara dan video atau gabungan dari semuanya yang disediakan melalui jalur koneksi internet sehingga dapat diakses dan dilihat oleh semua orang di seluruh dunia. Halaman *website* dibuat menggunakan bahasa standard yaitu HTML. Skrip HTML ini akan diterjemahkan oleh *web browser* sehingga dapat ditampilkan dalam bentuk informasi yang dapat dibaca oleh semua orang.

Dari definisi diatas penulis menyimpulkan bahwa *web* adalah sebuah kumpulan halaman pada suatu domain di internet yang dibuat dengan tujuan tertentu dan saling berhubungan serta dapat diakses secara luas menggunakan sebuah *browser*, yang berisi informasi data digital baik berupa teks, gambar, animasi, suara dan video atau gabungan dari semuanya yang disediakan melalui jalur koneksi internet

2.3.5 Pengertian Aplikasi Pendaftaran Investor pada Koperasi Konsumen Warmart Veteran Utama Palembang Berbasis *Web*

Aplikasi Pendaftaran Investor pada Koperasi Konsumen Warmart Veteran Utama Palembang adalah sistem aplikasi yang memanfaatkan *web* untuk melakukan pendaftaran investor yang di gunakan oleh admin pada Koperasi Konsumen Warmart Veteran Utama Palembang untuk mengolah data calon investor. Aplikasi ini berfungsi mempermudah proses *input* dan *output* pendaftaran investor serta mempermudah calon investor untuk melakukan pendaftaran tanpa harus datang langsung ke kantor administrasi koperasi.

2.4 Teori Program

2.4.1 Pengertian Basis Data *(Database)*

Menurut Kristanto (2018:79), "Basis data adalah kumpulan data, yang dapat digambarkan sebaagai aktivitas dari satu atau lebih organisasi yang berelasi". Adapun pengertian Basis Data menurut Conolly & Begg (dikutip Rusmawan, 2019:38), "Basis data adalah sebuah kumpulan data yang saling berelasi secara logika dan dirancang untuk memenuhi informasi yang dibutuhkan oleh suatu organisasi". Lain halnya menurut Abdullah (2018:103), "*Database* atau basis data adalah kumpulan informasi yang disimpan dalam computer secara sistematik sehingga dapat diperiksa menggunakan suatu program computer untuk memperoleh informasi".

Dari beberapa pernyataan diatas, dapat disimpulkan bahwa basis data adalah kumpulan data yang berelasi untuk memenuhi informasi yang tersimpan di computer secara sistematik supaya bisa di pantau dengan program komputer.

2.4.2 Pengertian *MySQL*

Menurut Kadir (dalam Pranata dkk. 2015:25), "*MySQL* merupakan *software* yang berbasis *structure query language* (SQL) tergolong sebagai DBMS (*Database Management System*) yang bersifat *Open Source. MySQL* adalah sebuah implementasi dari sistem manajemen *database* relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial." Sedangkan menurut Nugroho (2014:31), *"MySQL* atau program aplikasi *database*, yaitu *software* yang dapat kita pakai untuk menyimpan data berupa informasi *text* dan juga angka."

Dapat disimpulkan bahwa *MySQL* merupakan *software* DBMS yang bersifat *Open Source* dan digunakan untuk menyimpan data atau server *database* yang mendukung Bahasa *database* pencarian *SQL*.

2.4.3 Pengertian *CodeIgniter*

Sidik (dalam Destiningrum dan Adrian (2017:32), "*CodeIgniter* adalah Sebuah framework php yang bersifat *open source* dan menggunakan metode MVC *(Model, View, Controller)* untuk memudahkan developer atau programmer dalam membangun sebuah aplikasi berbasis web tanpa harus membuatnya dari awal".

Adapun menurut Hidayatullah dan Kawistara (2014:345), "*CodeIgniter* adalah suatu *framework* dengan metode *pattern Model View Controller* (MVC)". Hidayatullah dan Kawistara (2014:283-284) menjelaskan bahwa, "MVC adalah metode yang memisahkan *data logic (model)* dari *presentation logic (view) dan process logic (controller)."*

Sehingga dapat disimpulkan *CodeIgniter* adalah *framework* yang bersifat *open source* untuk memudahkan dalam membangun aplikasi tanpa harus membuat dari awal serta mapu memisahkan antara logika data dengan logika tampilan dan logika proses.

2.4. Bahasa Pemrograman

2.4.4.1 Pengertian HTML *(Hypertext Markup Language)*

Madcoms (dalam Pranata dkk. 2015:26), "HTML (HyperText Markup Language) dikenal sebagai bahasa kode berbasis teks untuk membuat sebuah halaman *web*, keberadaannya dikenal dengan adanya ekstensi *.htm atau *.html." Adapun menurut Supriyanto (dikutip Pranata dkk. 2015:26), "HTML merupakan suatu bahasa dari *website* (www) yang dipergunakan untuk menyusun dan membentuk dokumen agar dapat ditampilkan pada program *browser*." Lain hal nya Abdulloh (2018:127) meyatakan bahwa, "HTML merupakan singkatan dari *Hypertxt Markup Language* yaitu bahasa standar web yang dikelola penggunaanya oleh W3C *(Word Wide Web Consortium)* berupa tag-tag yang menyusun setiap elemen dari website".

2.4.4.2 Pengertian PHP *(Hypertext Preprocessor)*

Kadir (dikutip Pranata dkk. 2015:26), “PHP merupakan singkatan dari "PHP: *Hypertext Preprocessor*" adalah skrip yang dijalankan di *server*. Hasilnyalah yang dikirimkan ke klien, tempat pemakai menggunakan *browser*. Keuntungan PHP, kode yang menyusun program tidak perlu diedarkan ke pemakai sehingga kerahasiaan kode dapat dilindungi.” Sedangkan menurut Abdulloh (2018:127), *PHP* merupakan kependekan dari *PHP Hypertext preprocessor* yaitu bahasa pemrograman *web* yang dapat disisipkan dalam skrip *HTML* dan bekerja di sisi server.” Adapun menurut pendapat Raharjo (2016:38), “PHP adalah salah satu bahasa pemrograman skrip yang dirancang untuk membangun aplikasi web”.

2.4.4.3 Pengertian *Web Server*

Abdulloh (2018:4), “*Web server* adalah program untuk melayani penyajian halaman *web* di internet. Untuk kebutuhan simulasi, anda dapat menginstal *software web server* di komputer local. Jika anda hanya mendesain *web* statis menggunakan *HTML* dan *CSS*, maka anda tidak membutuhkannya. Namun jika anda mendesain web dinamis menggunakan script pemrograman seperti *PHP*, maka anda membutuhkannya.

Program *web server* popular adalah *Apache*.anda dapat menginstal paket softwarenya sesuai system operasi anda. Misalnya untuk windows anda bias memakai WAMP atau XAMPP yang berisi paket *software Apache*, *MySQL*, dan *PHP*.”

2.4.4.4 Pengertian XAMPP

Suntoro (2019:11), “XAMPP adalah perangkat lunak yang bersifat *open source*, aplikasi *Apache* (web server) yang mudah di instal dan berisi MariaDB, PHP, dan Perl”. Sedaangkan, menurut Santoso dan Nurmalina (2017:86), Xampp merupakan alat bantu yang menyediakan paket perangkat lunak ke dalam satu buah paket. Dengan menginstal XAMPP maka tidak perlu lagi melakukan instalasi dan konfigurasi *web server Apache*, *PHP* dan MySQL secara manual. XAMPP akan menginstalasi dan mengkonfigurasikannya secara otomatis atau

auto konfigurasi. XAMPP merupakan paket *PHP* yang berbasis *open source* yang dikembangkan oleh sebuah komunitas *Open Source*. Dengan menggunakan XAMPP tidak dibingungkan dengan penginstalan program-program lain, karena semua kebutuhan telah tersedia oleh XAMPP. Yang terdapat pada XAMPP di antaranya : *Apache*, MySQL, *PHP*, FilZilla FTP *Server*, PhpMyAdmin dll.

2.4.4.5 Pengertian *phpMyAdmin*

Nugroho (2014:10), phpMyAdmin adalah aplikasi manajemen *database server* MySQL berbasis *web*. Dengan aplikasi phpMyAdmin kita bias mengelola *database* sebagai *root* (pemilik server) juga disebut *Administrator database.* Lewat fasilitas phpMyAdmin, Anda bias membuat *database* baru, *table* dan struktur data mengelola data di dalam *database*.