BAB II
TINJAUAN PUSTAKA

2.1 Teori Umum

2.1.1 Pengertian Komputer

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

Rosdiana (2016:1), "Komputer diterjemahkan sebagai sekumpulan alat elektronik yang saling bekerja sama, dapat menerima data (*input*), mengolah data (proses) dan memberi informasi (*output*) serta terkoordinasi dibawah kontrol program yang tersimpan di memorinya."

Kesimpulannya, Komputer adalah alat bantu pemrosesan data secara elektronik yang bermanfaat untuk melaksanakan berbagai pekerjaan manusia.

2.1.2 Pengertian Perangkat Lunak

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

Kadir (2017:2), "Perangkat Lunak adalah instruksi-instruksi yang ditujukan kepada komputer agar dapat melaksanakan tugas sesuai kehendak pemakai"

Kesimpulannya, Perangkat Lunak adalah kumpulan beberapa intruksi yang diperintahkah dan diproses dengan bantuan mesin komputer yang terasosiasi dengan dokumentasi perangkat lunak seperti dokumentasi kebutuhan.

2.1.3 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."

Kesimpulannya, Sistem adalah sekumpulan prosedur yang saling berkaitan dan saling terhubung satu sama lain dan bekerja sama untuk mencapai suatu tujuan.

2.1.4 Pengertian Basis Data

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".

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

Kesimpulannya, Basis data adalah kumpulan data yang disimpan secara bersamaan sedemikian rupa dan membuat informasi tersedia untuk memenuhi berbagai kebutuhan.

2.1.5 Metode Pengembangan Aplikasi

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

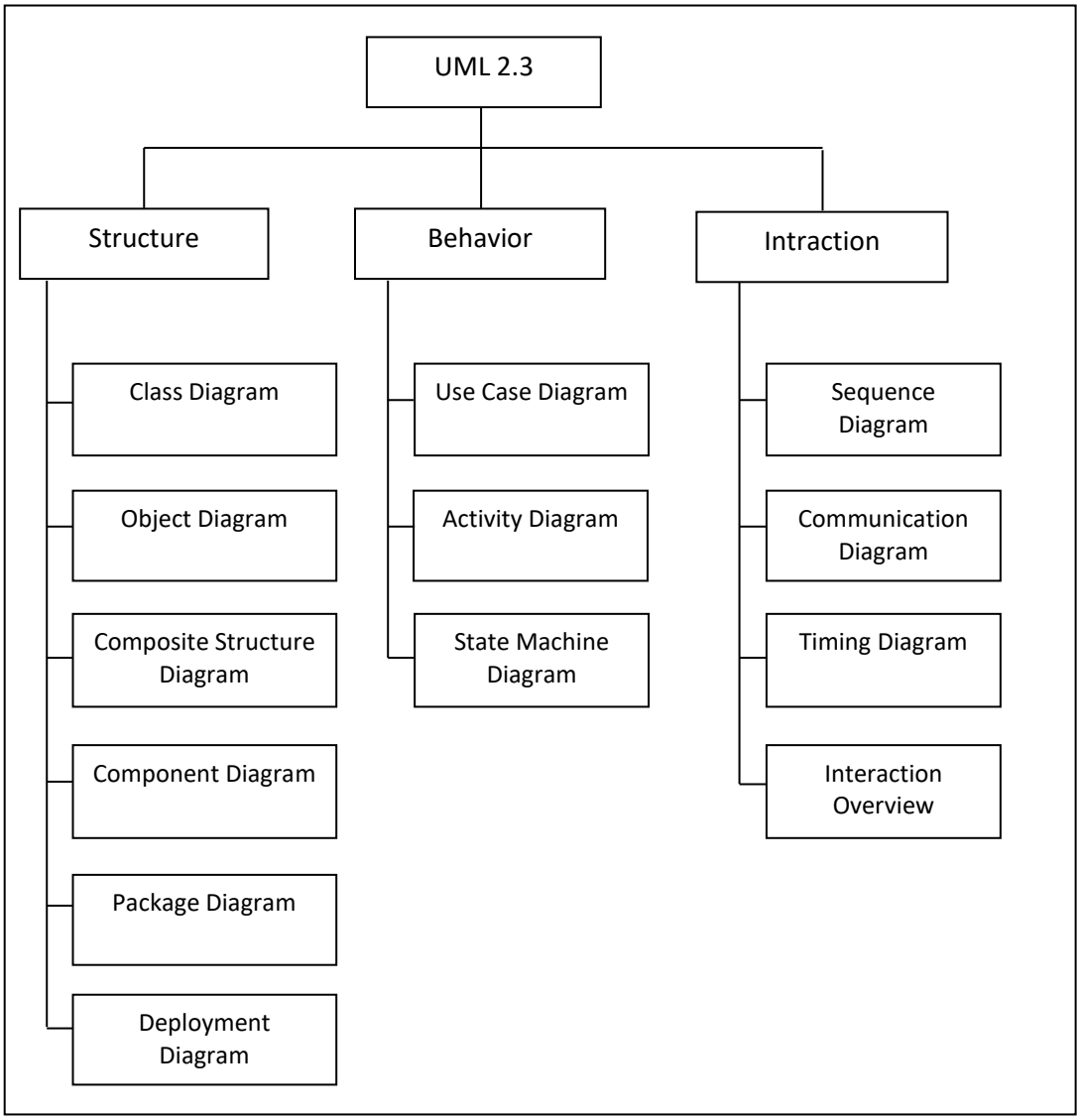
4. *Transition* (transisi)

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 Khusus

2.2.1 Pengertian *UML (Unified Modeling Language)*

Menurut Sukamto dan Shalahuddin (2016:140), “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:



Sumber : Sukamto dan Shalahuddin (2016:141)

Gambar 2.1 Macam-macam Diagram UML

Penjelasan singkat dari pembagian kategori pada diagram UML menurut Sukamto dan Shalahuddin (2016:141) :

1) *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2) *Behavior diagrams* yaitu kumpulam diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahana yang terjadi pada sebuah sistem.

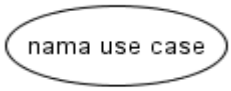
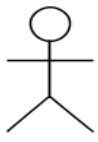
3) *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

2.2.2 Pengertian Use Case Diagram

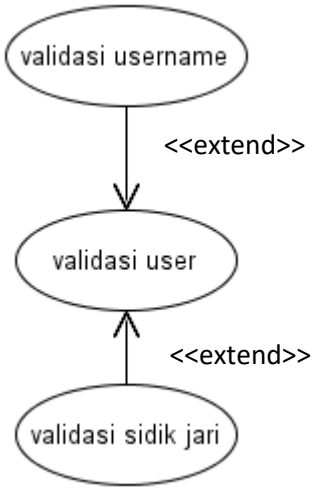
Sukamto dan Shalahuddin (2018:155), menjelaskan tentang *use case* diagram sebagai berikut :

"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:Berikut simbol-simbol pada Use Case Diagram :

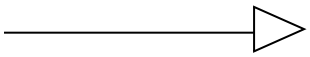
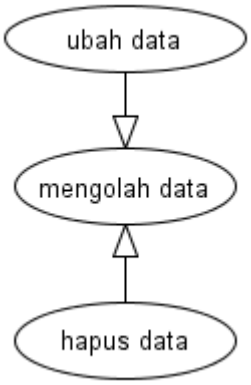
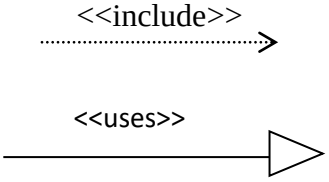
Tabel 2.1 Simbol-simbol pada *Use case* Diagram

No	Simbol	Deskripsi
1	*Use case* 	fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal-awal frase nama *use case*
2	aktor / *actor* 	orang, proses, atau siatem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor
3	asosiasi / *association* ———————	komunikasi antar aktor dan *use case* yang berpartisioasi pada *use case*.

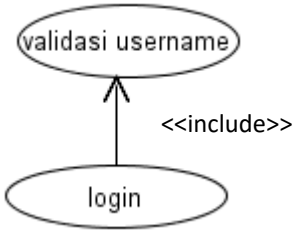
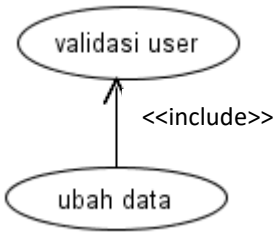
Lanjutan Tabel 2.1 Simbol-simbol pada *Use case* Diagram

No	Simbol	Deskripsi
4	ekstensi / *extend* <<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

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

No	Simbol	Deskripsi
5	Generalisasi / *generalization* 	hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya,  misalnya: arah panah mengarah pada *use case* yang menjadi generalisasinya (umum)
6	menggunakan / 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:

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

No	Simbol	Deskripsi
		 *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 (2018:155)

Ada dua hal utama pada *use case* yaitu:

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

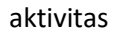
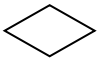
2.2.3 Pengertian *Activity* Diagram

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

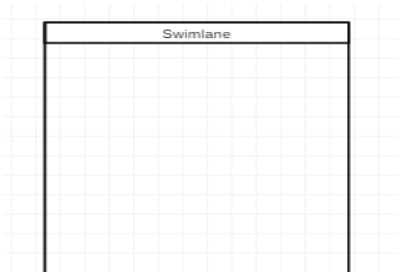
"A*ctivity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem."

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

Tabel 2.2 Simbol-simbol pada Activity Diagram

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

Lanjutan Tabel 2.2 Simbol-simbol pada *Activity* Diagram

No	Simbol	Deskripsi
6	Swimlane  atau 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

Sumber : Sukamto dan Shalahuddin (2018:161)

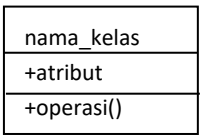
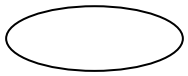
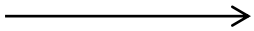
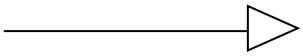
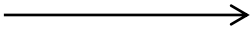
2.2.4 Pengertian Class Diagram

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 2.3 Simbol-simbol pada Class Diagram

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

Sumber : Sukamto dan Shalahuddin (2018:141)

2.2.5 Pengertian *Sequence Diagram*

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

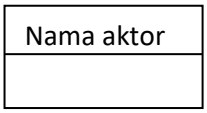
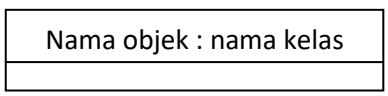
"Diagram sekuen menggambarkan kelakuan objek pada use case dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima

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".

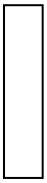
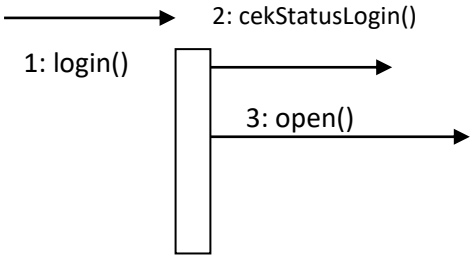
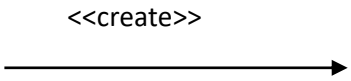
Banyaknya diagram sekuen yang harus digambarkan adalah minimal sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* yang didefinisikan maka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

Berikut simbol-simbol pada Sequence Diagram :

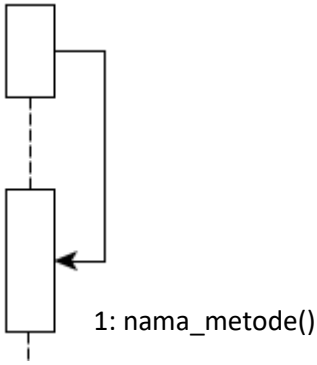
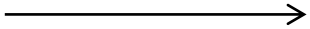
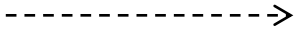
Tabel 2.4 Simbol-simbol pada *Sequence* Diagram

No	Simbol	Deskripsi
1	Actor  atau  tanpa waktu aktif	orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal frase nama actor
2	Garis hidup / *lifeline* 	menyatakan kehidupan suatu objek
3	 Objek	menyatakan objek yang berinteraksi pesan

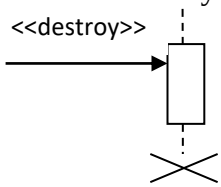
Lanjutan Tabel 2.4 Simbol-simbol pada *Sequence* Diagram

No	Simbol	Deskripsi
4	Waktu aktif 	menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya  maka cekStatusLogin () dan open() dilakukan di dalam metode login() aktor tidak memiliki waktu aktif
5	Pesan tipe create 	menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat

Lanjutan Tabel 2.4 Simbol-simbol pada *Sequence* Diagram

No	Simbol	Deksripsi
	Pesan tipe call <<create>> 	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
6	Pesan tipe send 1: masukan 	menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi
7	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

Lanjutan Tabel 2.4 Simbol-simbol pada *Sequence* Diagram

No	Simbol	Deskripsi
	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 (2018:141)

2.2.6 Pengertian Kamus Data

Sukamto dan Shalahuddin (2018:73) menjelaskan, "Kamus data adalah kumpulan daftar elemen data yang mengalir pada sistem perangkat lunak sehingga masukkan (input) dan keluaran (ouput) dapat dipahami secara umum (memiliki standar cara penulisan)." Kamus data memiliki beberapa simbol sebagai berikut :

Tabel 3.6 Simbol-simbol Kamus Data

Simbol	Keterangan
=	Disusun atau terdiri dari
+	Dan
[|]	Baik...atau...
$\{ \}^n$	n kali diulang/bernilai banyak
( )	Data opsional
..	Batas komentar

Sumber : Sukamto dan Shalahuddin (2018:73)

2.3 Teori Judul

2.3.1 Pengertian Aplikasi

Indrajani (2018:3), menjelaskan bahwa "Aplikasi adalah program yang menentukan aktivitas pemrosesan informasi yang dibutuhkan untuk penyelesaian tugas-tugas khusus dari pemakai komputer."

Sutarman (2013:10), menjelaskan bahwa "program aplikasi adalah program-program yang dibuat oleh suatu perusahaan komputer untuk para pemakai yang beroperasi dalam bidang-bidang umum, seperti toko, penerbitan, komunikasi, penerbangan, perdagangan, dan sebagainya."

Kesimpulannya, aplikasi atau program aplikasi adalah *software* atau program-program yang dibuat untuk membantu para pemakai atau *user* dalam mengerjakan tugas-tugas tertentu.

2.3.2 Pengertian Absensi

Santoso dan Nurmalina (2017:84) menjelaskan bahwa "Absensi adalah suatu cara untuk mengetahui sejauh mana tingkat disiplin kerja, apakah orang yang bekerja mampu menanti peraturan yang berlaku."

Setiawan dan Kurniawan (2015:44), "Absensi adalah suatu pendataan kehadiran yang merupakan bagian dari aktifitas pelaporan yang ada dalam sebuah institusi."

Kesimpulannya, Absensi adalah aktifitas pelaporan yang ada dalam sebuah institusi yang berkaitan dengan pendataan kehadiran dan ketidakhadiran.

2.3.3 Pengertian Pegawai

Sitompul (2015:23) Pegawai merupakan Anggota suatu usaha kerjasama (organisasi) dengan maksud memperoleh balas jasa atau imbalan kompensasi atas jasa yang telah diberikan.

Pasal 1 Undang-undang Nomor 43 Tahun 1999 memberikan pengertian Pegawai Negeri adalah setiap warga negara Republik Indonesia yang telah memenuhi syarat yang ditentukan, diangkat oleh pejabat berwenang dan diserahi tugas dalam suatu jabatan negeri.

Kesimpulannya, Pegawai adalah orang yang bekerja pada suatu lembaga pemerintahan yang diangkat oleh pejabat berwenang dan diserahi tugas dalam suatu jabatan negeri.

2.3.4 Pengertian *Geotagging*

Halili MZ (2017:02), "*Geotag* atau *Geotaging* merupakan suatu proses penambahan informasi geografis pada berbagai macam media, seperti foto, video, website, dan jejaring sosial."

Wijaya, dkk (2018:03), *Geotagging* merupakan proses penambahan informasi *geospasial* pada berbagai media digital.

Kesimpulannya, *Geotagging* merupakan suatu proses penambahan informasi geografis pada berbagai media digital.

2.3.5 Pengertian Aplikasi Absensi Pegawai dengan Geotagging Pada Dinas Penanaman Modal dan Pelayanan Terpadu Satu Pintu Provinsi Sumatra Selatan berbasis *Android Webservice*

Aplikasi Absensi Pegawai dengan Geotagging Pada Dinas Penanaman Modal dan Pelayanan Terpadu Satu Pintu Provinsi Sumatra Selatan berbasis *Android Webservice* adalah aplikasi yang dibuat untuk dapat membantu pegawai dalam melakukan kegiatan absensinya serta memudahkan instansi dalam mengolah data absensi tersebut dengan dilengkapi koordinat lokasi pegawai saat melakukan absensi.

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 (2014:186), " *XAMPP* adalah sebuah paket kumpulan software yang terdiri dari Apache, *MySQL, PhpMyAdmin, Perl, Filezilla* dan lain-lain."

Kesimpulannya, XAMPP adalah program web lengkap yang terdiri dari *server 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 *multitheard, multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia."

Kesimpulannya, MySQL adalah bahasa yang digunakan untuk mengelola data SQL (database management system) atau DBMS yang *multitheard, multi-user.*

2.4.3 Pengertian *PHPMyAdmin*

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

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".*

Jadi, *PHPMyAdmin* adalah aplikasi yang digunakan untuk membuat *database* secara cepat dan mudah, mendukung berbagai aktivitas *MySQL* seperti pengelolaan data, *table*, relasi antar tabel, dan lain sebagainya.

2.4.4 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."

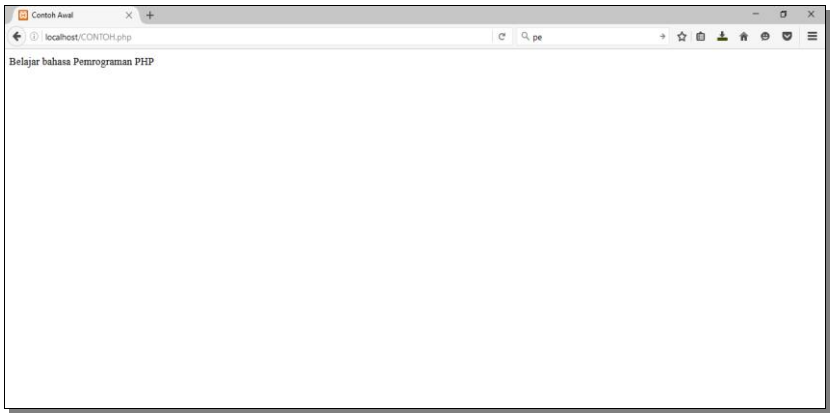
Kesimpulannya, PHP adalah bahasa pemograman berbasis *server-side* yang bisa kita gunakan untuk membuat aplikasi web yang disisipkan pada HTML.

Gambar 2.2. *Tampilan Logo PHP*

2.4.4.1 Sintaks Dasar *PHP*

Kode (Script) PHP yang sering disebut dengan istilah embedded script yaitu script PHP yang disisipkan di antara sricpt HTML. Jadi dapat dikatakan script PHP hanya ditulis atau disisipkan ketika dibutuhkan saja, seperti menampilkan data dari database meng-upload file, delete data, edit data dan lain sebagainya. Contoh script :

```
<HTML>
    <HEAD>
        <TITLE>Contoh Awal</TITLE>
    </HEAD>
<BODY>
    <?php
        echo “Belajar bahasa Pemrograman PHP”;
    ?>
    </BODY>
</HTML>
```



Gambar 2.3. Contoh *Script PHP*

2.4.4.2. 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 variable yang diset menjadi tipe data Null, ini menjadikan variabel tersebut kosong.

8. **Resources** tipe data spesial yang satu ini dikhususkan untuk menyimpan *resources*, sumber atau alamat.

2.4.5 Pengertian *Android*

Juhara (2016 : 1), android adalah sistem operasi berbasis linux yang dimodifikasi untuk perangkat bergerak (*mobile device*) yang terdiri dari sistem operasi, middleware, dan aplikasi-aplikasi utama.

Safaat H (2015 : 1), android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi.

Kesimpulannya, Android adalah sistem operasi berbasis linux yang dimodifikasi untuk perangkat bergerak (*mobile device)* yang mencakup sistem operasi, *middleware* dan aplikasi.

2.4.6 Pengertian *Java*

Jubilee Enterprise (2016:1), "Java merupakan Bahasa pemrograman yang sangat populer karena rentang aplikasi yang bisa dibuat menggunakan bahasa ini sangatlah luas, mulai dari komputer hingga smartphone."

Hariyanto (2014:3), Java merupakan Bahasa orientasi objek untuk pengembangan aplikasi mandiri, aplikasi berbasis internet, aplikasi untuk perangkat cerdas yang dapat berkomunikasi lewat internet/jaringan komunikasi.

Kesimpulannya, Java adalah Bahasa pemrograman yang sangat populer untuk pengembangan aplikasi mandiri, aplikasi berbasis internet, aplikasi untuk perangkat cerdas yang dapat berkomunikasi lewat internet/jaringan komunikasi.