# BAB II
# TINJAUAN PUSTAKA

## 2.1 Teori Umum

### 2.1.1 Komputer

Menurut (V. C. Hamacher, Z. G. Vranesic. S. G. Zaky) mengemukakan bahwa, “komputer adalah mesin penghitung elektronik yang cepat dan menerima informasi input digital, memprosesnya sesuai dengan suatu program yang tersimpan dimemorinya (stored program) dan menghasilkan output informasi.”

Menurut Sanders (1985) mengemukakan bahwa, “komputer adalah sistem elektronik untuk memanipulasi data yang cepat dan tepat serta dirancang dan diorganisasikan supaya secara otomatis menerima dan menyimpan data input, memprosesnya, dan menghasilkan output berdasarkan instruksi-instruksi yang telah tersimpan di dalam memori.”

### 2.1.2 Perangkat Lunak

Menurut Roger (2002:10) mengemukakan bahwa, “Perangkat lunak atau software adalah sebuah perintah program dalam sebuah komputer, yang apabila diberi perintah oleh user akan memberikan fungsi dan unjuk kerja seperti yang diharapkan oleh usernya. Pernyataan ini menggambarkan bahwa software atau perangkat lunak ini berfungsi memberi perintah untuk komputer, agar komputer berfungsi secara optimal, sesuai dengan kemauan user atau pengguna yang memberikan perintah..”

Menurut Melwin (2007), mendefinisikan perangkat lunak sebagai berikut, “Perangkat lunak Berfungsi sebagai pengatur aktivitas kerja komputer dan semua intruksi yang mengarah pada sistem komputer. Perangkat lunak menjembatani interaksi user dengan komputer yang hanya memahami bahasa mesin.” Software dibangun berdasarkan permintaan atau kebutuhan penggunanya. Ini sangat jelas pada *software* aplikasi.”

### 2.1.3 Data

Menurut Indrajani (2018:2), “Data adalah fakta atau observasi mentah yang biasanya mengenai fenomena fisik atau transaksi data.”

Menurut Fathansyah (2018:2), “Data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia(pegawai, siswa, pembeli, pelanggan),barang, hewan, peristiwa, konsep, keadaan, dan sebagainya, yang diwujudkan dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya.”

### 2.1.4 Basis Data

Menurut Faridi, dkk dalam Jurnal CERITA (2016:192), mendefinisikan bahwa “Database adalah sebuah struktur yang umumnya dikategorikan dalam 2 hal: sebuah database flat dan sebuah database relasional. Database relasional lebih disukai karena lebih masuk akal dibandingkan database flat”.

Sedangkan Menurut Anhar (2016:19), “Database (basis data) dapat diartikan sebagai suatu pengorganisasian data dengan bantuan komputer, yang memungkinkan dapat diakses dengan mudah dan cepat”.

### 2.1.5 Sistem

Menurut McLeod dalam buku Yakub (2012:01) sistem adalah “sekelompok elemen-elemen yang terintegrasi dengan tujuan yang sama untuk mencapai tujuan”.

Menurut Gordon dalam buku Ais Zakiyudin (2011:01) mendefinisikan sistem sebagai “seperangkat unsur-unsur yang terdiri dari manusia, alat, konsep dan prosedur yang di himpun menjadi satu untuk maksud dan tujuan bersama”.

Menurut Jerry dalam buku Jogiyanto (2010:90) sistem adalah “suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau melakukan suatu sasaran tertentu”.

### 2.1.5.1 Karakteristik Sistem

Menurut Hartono (2013:14) menyatakan bahwa sebuah sistem memiliki paling sedikit sepuluh karakteristik sebagai berikut:

1. **Komponen (components)**

   Bagian-bagian atau elemen-elemen yang dapat berupa benda atau manusia, berbentuk nyata atau abstrak, dan disebut subsistem.

2. **Penghubung antarbagian (interface)**

   Sesuatu yang bertugas menjembatani satu bagian dengan bagian lain, dan memungkinkan terjadinya interaksi/komunikasi antarbagian.

3. **Batas (boundary)**

   Sesuatu yang membedakan antara satu sistem dengan sistem atau sistem-sistem lain.

4. **Lingkungan (environment)**

   Segala sesuatu yang berada diluar sistem dan dapat bersifat menguntungkan atau merugikan sistem yang bersangkutan.

5. **Masukan (input)**

   Sesuatu yang merupakan bahan untuk diolah atau diproses oleh sistem.

6. **Mekanisme pengolahan (processing)**

   Perangkat dan prosedur untuk mengubah masukan menjadi keluaran dan menampilkannya.

7. **Keluaran (output)**

   Berbagai macam bentuk hasil atau produksi yang dikeluarkan dari pengolahan.

8. **Tujuan (goal/objective)**

   Sesuatu atau keadaan yang ingin dicapai oleh sistem, baik dalam jangka pendek maupun jangka panjang.

9. **Sensor dan kendali (sensor & control)**

   Sesuatu yang bertugas memantau dan menginformasikan perubahan-

perubahan di dalam lingkungan dan dalam diri sistem kepada sistem.

**10. Umpan-balik (feedback)**

Informasi tentang perubahan-perubahan lingkungan dan perubahan-perubahan (penyimpangan) dalam diri sistem.

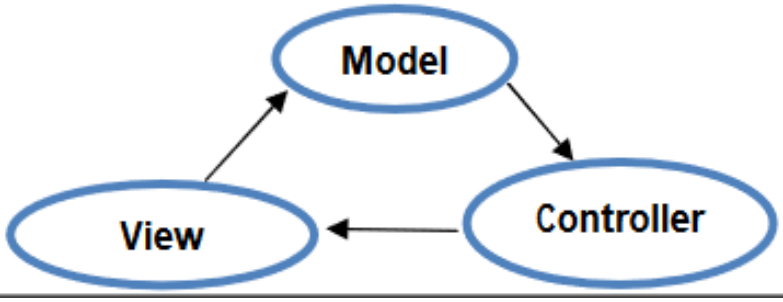
### 2.1.6 *Framework*

Menurut Sidik (2012 : 1), dengan menggunakan framework, kita tidak perlu membuat program dari awal, tetapi kita sudah diberikan library fungsi-fungsi yang sudah diorganisasikan untuk dapat membuat suatu program dengan cepat.

Menurut Rosa dan Shalahuddin (2011 : 191), framework merupakan kerangka kerja yang memudahkan programmer untuk membuat sebuah aplikasi sehingga programmer akan lebih mudah melakukan perubahan (customize) terhadap aplikasinya dan dapat memakainya kembali untuk aplikasi lain yang sejenis.

### 2.1.7 *Model View Controller* (MVC)

Menurut Sidik (2012 : 29 - 30), Model View Controller (MVC) merupakan teknik pemrograman yang popular saat ini, yang mengharapkan pemrogram secara disiplin untuk membagi program menjadi tiga bagian yaitu, model, view dan controller, seperti gambar berikut :




**Gambar 2.1** Bagian MVC

1. Model : Merupakan bagian dari aplikasi yang mengimplementasi logika untuk domain data aplikasi.
2. View : Merupakan komponen yang menampilkan antarmuka untuk pengguna (user interface) aplikasi.
3. Controller : Merupakan komponen yang digunakan untuk menangani interaksi pengguna, bekerja dengan model, dan memilih view mana yang digunakan untuk merender data.

Menurut Rosa dan Shalahuddin (2011 : 192), konsep Model View Controller (MVC) bertujuan agar sebuah aplikasi dapat mudah diplihara oleh orang-orang di dalam tim pengembangan yang berbeda spesifikasi pekerjaan, misalnya database administrator (DBA) untuk mengurusi masalah basis data, blok controller untuk programmer, dan blok view untuk desainer antarmuka (interface designer).

### 2.1.8 Metode Pengembangan Aplikasi

Penelitian ini menggunakan metode pengembangan perangkat lunak dengan RUP (*Rational Unified Process*). Menurut Rosa dan Shalahuddin (2014:125), "RUP (*Rational Unified Process*) merupakan 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 Rosa 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 Pemrograman Berorientasi Objek Oriented (OOP)

Menurut Adi Sulistyo (2018:97) dalam bukunya mengatakan bahwa, “Pemrograman berorientasi objek atau *object-oriented programming* (OOP) merupakan paradigma pemrograman yang berorientasikan kepada objek.”

Menurut Ramadhani (2015) mengemukakan bahwa, “Pemrograman berorientasi objek atau object oriented programming (OOP) merupakan suatu pendekatan pemrograman yang menggunakan object dan class. OOP

memberikan kemudahan dalam pembuatan sebuah program", keuntungan yang didapat apabila membuat Program berorientasi objek atau object oriented programming (OOP) antara lain :

1) Reusability, kode yang dibuat dapat digunakan kembali
2) Extensibility , pemrogram dapat membuat methode baru atau mengubah yang sudah ada sesuai yang diinginkan tanpa harus membuat kode dari awal
3) Maintainability, kode yang sudah dibuat lebih mudah untuk dikelola apabila aplikasi yang dibuat berskala besar yang memungkinkan adanya error dalam pengembangannya hal tersebut dapat diatasi dengan OOP karena pemrograman OOP sudah menggunakan konsep modularitas

### 2.2.2 Unified Modeling Language(UML )

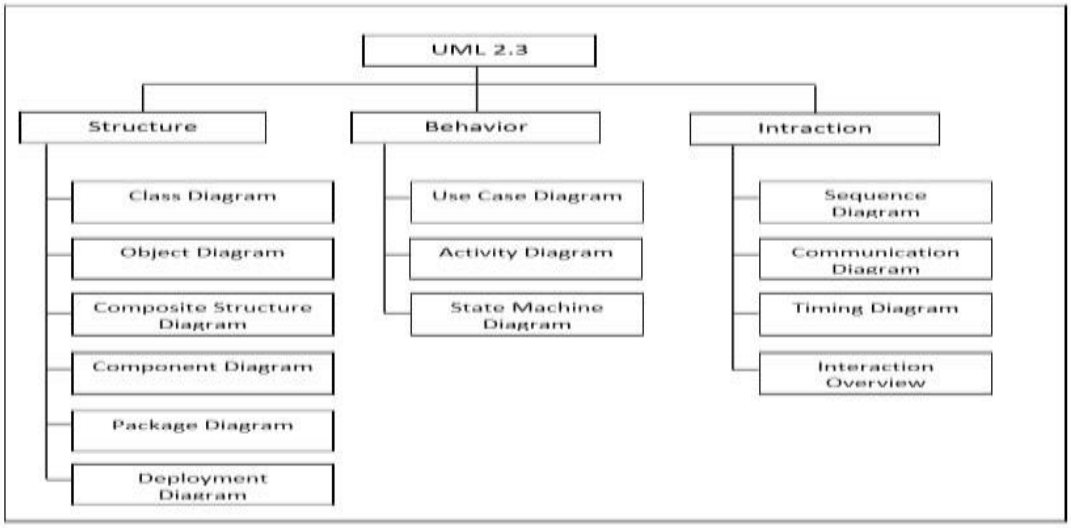
Menurut Munawar (2018:49) mengemukakan bahwa, "*Unified Modeling Language* (UML) adalah salah satu alat bantu yang sangat handal di dunia pengembangan system yang berorientasi objek. Hal ini disebabkan karena UML menyediakan Bahasa Pemodelan Visual yang memungkinkan bagi pengemban system untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti serta dilengkapi dengan mekanisme yang efektif untuk berbagi (sharing) dan mengkomunikasikan rancangan mereka dengan yang lain."

Menurut (Sukamto & Shalahuddin 2016), "UML (Unified Modeling Language) adalah salah standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan requirement, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek".

### 2.2.3 Macam-macam Diagram 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** Macam-macam Diagram UML

*Sumber : Sukamto dan Shalahuddin (2018:155)*

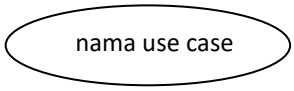
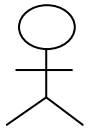
Penjelasan singkat dari pembagian kategori pada diagram UML menurut Sukamto dan Shalahuddin (2018:141):

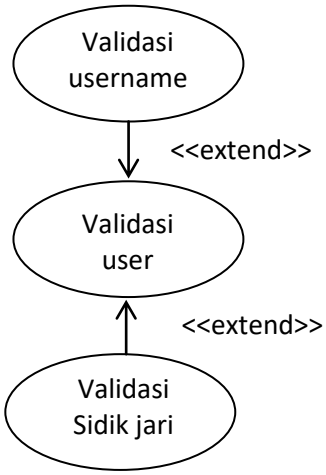
1) *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.
2) *Behavior diagrams* yaitu kumpulam diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahana yang terjadi pada sebuah sistem.
3) *Interaction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

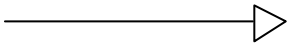
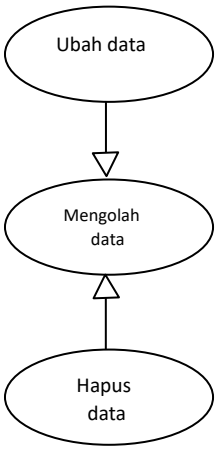
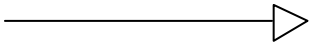
#### 2.2.3.1 *Use Case Diagram*

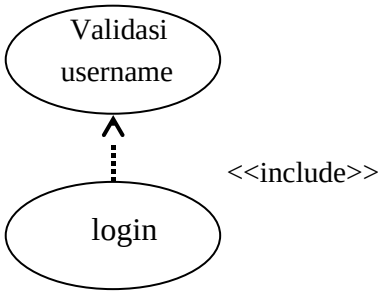
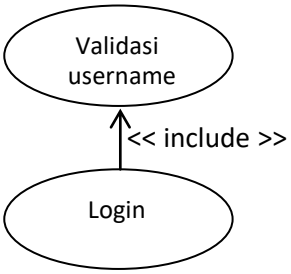
Menurut Munawar (2018:89) mengemukakan bahwa, ”*Use Case* adalah deskripsi fungsi dari sebuah *system* dari perspektif pengguna. *Use case* bekerja dengan cara mendeskripisikan tipikal interaksi antara *user* sebuah system dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai.”

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

| No. | Simbol | Deskripsi |
|---|---|---|
| 1 | *Use Case*<br>  | fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awalawal frase nama use case. |
| 2 | Aktor / *Actor*<br>  <br>nama aktor | 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*<br>_____ | komunikasi antar aktor dan use case yang berpartisioasi pada use case. |

| No | Simbol | Deskripsi |
|---|---|---|
| 4 | *ekstensi / extend*<br><br><<extend>><br>  | 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<br>  <br>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. |

| 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*  <<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:<br>□ Include berarti use case yang ditambahkan akan selalu di panggil saat use case tambahan dijalankan, misal pada kasus berikut : |

| No | Simbol | Deskripsi |
|---|---|---|
| | |  <br>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:<br>  <br>kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan. |

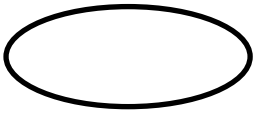
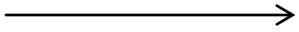
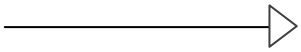
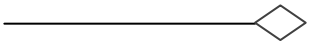
*Sumber : Sukamto dan Shalahuddin (2018:155)*

### 2.2.3.2 *Class Diagram*

Menurut Munawar (2018:101) mengemukakan bahwa, "*Class Diagram* adalah diagram statis. Ini mewakili pandangan statis dari suatu aplikasi. *Class diagram* tidak hanya digunakan untuk memvisualisasikan, menggambarkan, dan

mendokumentasikan berbagai aspek system, tetapi juga untuk membangun kode eksekusi (*executable code*) dari aplikasi perangkat lunak."

**Tabel 2.2** Simbol-simbol pada *Class Diagram*

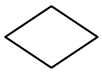
| No | Simbol | Deskripsi |
|---|---|---|
| 1 | Kelas<br>nama_kelas<br>+atribut<br>+operasi() | Kelas pada struktur sistem. |
| 2 | antarmuka / interface<br>  <br>nama_interface | Sama dengan konsep interface dalam pemrograman berorientasi objek. |
| 3 | Asosiasi / association<br>——— | Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai multiplicity |
| 4 | Asosiasi berarah / directed association<br>  | Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity |
| 5 | Generalisasi<br>  | Relasi antarkelas dengan makna generalisasi – spesialisasi (umum khusus) |
| 6 | Ketergantungan / dependency<br>  | Relasi antarkelas dengan makna kebergantungan antar kelas |
| 7 | Agreisasi/agregation<br>  | Relasi antarkelas dengan makna semua-bagian (whole-part) |

*Sumber : Sukamto dan Shalahuddin (2018:141)*

### 2.2.3.3 *Activity Diagram*

Menurut Munawar (2018:127) mengemukakan bahwa, "*Activity Diagram* adalah bagian terpenting dari UML yang menggambarkan aspek dinamis dari system. Logika prosudural, proses bisnis dan aliran kerja suatu bisnis bisa dengan mudah dideskripsikan dalam *activity diagram*. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku parallel sedangkan *flowchart* tidak bisa".

**Tabel 2.3** Simbol-simbol pada *Activity Diagram*

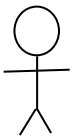
| No. | Simbol | Deskripsi |
|---|---|---|
| 1 | Status awal  | Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal |
| 2 | Aktivitas <br> aktivitas | Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja |
| 3 | Percabangan / *decision*  | Asosiasi percabangan di mana jika ada pilihan aktivitas lebih dari satu |
| | 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 . |

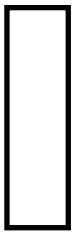
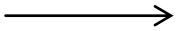
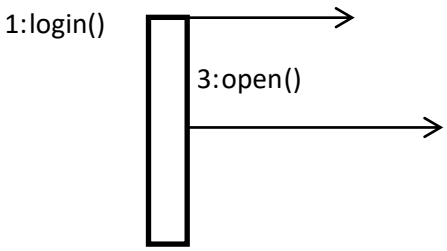
*Sumber : Sukamto dan Shalahuddin (2018:161)*

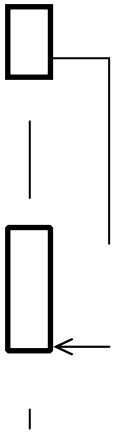
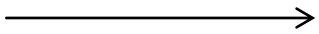
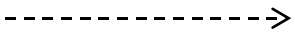
### 2.2.3.4 *Sequence Diagram*

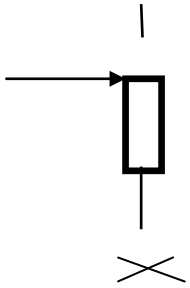
Menurut Munawar (2018:137) mengemukakan bahwa, "*Sequence diagram* digunakan untuk menggambarkan perilaku pada sebuah scenario. Diagram ini menunjukan sejumlah contoh *obyek* dan *message* (pesan) yang diletakkan diantara obyek-obyek ini di dalam *usecase*."

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

| No | Simbol | Deskripsi |
|---|---|---|
| 1 | Actor  Nama Actor<br>Atau<br>Nama Actor<br>Tanapa 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, biasaynya dinyatakan kata benda diawal frase nama aktor. |
| 2 | Garis hidup/lifeline  | Menyatakan kehidupan suatu objek. |
| 3 | Nama objek : nam kelas<br>Objek | Menyatakan objek yang berinteraksi pesan |

| | | |
|---|---|---|
| 4 | Waktu aktif<br>  | menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan di dalamnya, misalnya<br>  <br>2.cekStatusLogin()<br>  <br>maka cekStatusLogin () dan open() dilakukan di dalam metode login() aktor tidak memiliki waktu aktif |
| 5 | Pesan tipe create<br><<create>><br>  | menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat |

| No | Simbol | Deskripsi |
|---|---|---|
| 6 | Pesan tipe call<br><br><<create>><br>  | menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri,<br>  <br>1:nama_metode()<br>arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi |
| 7 | Pesan tipe send<br>  <br>1: masukan | menyatakan bahwa suatu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi |
| 8 | Pesan tipe return<br><br>1: keluaran<br>  | menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian |

| | | |
|---|---|---|
| 9 | Pesan tipe destroy<br><<destroy>><br>  | 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)*

## 2.3 Pengertian Judul

### 2.3.1 Aplikasi

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

Menurut Pane dkk (2020:132) menjelaskan bahwa "Pengertian aplikasi adalah program yang dibuat dengan tujuan untuk melaksanakan fungsi sesuai dengan kegunaan aplikasinya, penggunanya, dan jenis aplikasi itu sendiri".

### 2.3.2 Monitoring

Menurut Muginoputro (1998), "Monitoring adalah pemantauan yang dapat dijelaskan sebagai kesadaran tentang apa yang ingin diketahui, pengalaman berkadar tingkat tinggi dilakukan agar dapat membuat pengukuran melalui waktu yang menunjukkan pergerakan ke arah tujuan atau menjauh dari itu".

Menurut KBBI (1999), "Monitoring adalah pemantauan yang dapat memberikan informasi tentang status dan kecenderungan bahwa pengukuran dan evaluasi yang diselesaikan berulang dari waktu ke waktu, pemantauan umumnya dilakukan untuk tujuan tertentu, untuk memeriksa terhadap proses berikut objek atau untuk mengevaluasi kondisi atau kemajuan menuju tujuan hasil manajemen

atas efek tindakan dari beberapa jenis antara lain tindakan untuk mempertahankan manajemen yang sedang berjalan".

### 2.3.3 Kondisi Tidak Aman (KTA)

Menurut Joko Priono (2018:21), "KTA merupakan suatu keadaan (umumnya tempat kerja) yang ada di sekitar kita yang memiliki potensi menyeabkan cidera atau kecelakaan kerja serta kerusakan lainnya".

### 2.3.4 Tindakan Tidak Aman (TTA)

Menurut Joko Priono (2018:23), "TTA adalah suatu perilaku membahayakan atau tidak aman yang dapat menyebabkan kecelakaan kerja menimbulkan kerugian cedera hingga kematian".

### 2.3.5 Pengertian PT. Bina Sarana Sukses Job Site PT. PMSS

PT. Bina Sarana Sukses adalah Perusahaan yang bergerak di bidang pertambangan yang terletal di Desa Pulau Panggung kecamatan Lawang Kidul Kabupaten Muara Enim, Sumatera Selatan dan memiliki beberapa Job Site di berbagai kota salah satunya di kota Palembang Sumatera Selatan. PT. Bina Sarana Sukses yang mengelola pertambangan batu bara.

### 2.3.6 Aplikasi Monitoring Kondisi Tidak Aman (KTA) dan Tindakan Tidak Aman (TTA) pada PT. Bina Sarana Sukses Job Site PT. PMSS

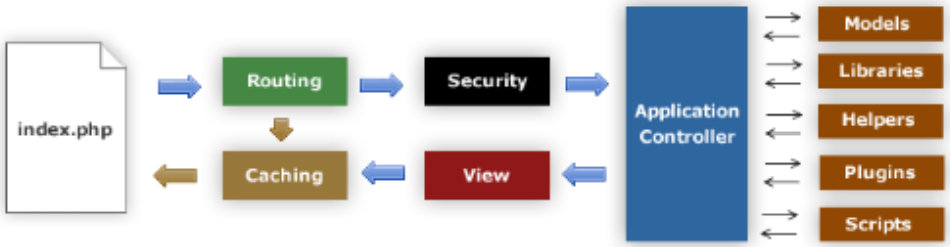
Aplikasi Monitoring Kondisi Tidak Aman (KTA) dan Tindakan Tidak Aman (TTA) pada PT. Bina Sarana Sukses Job Site PT. PMSS adalah suatu aplikasi yang berfungsi untuk mempermudah para karyawan di departemen Safety Healty Environment (SHE) dalam input dan moniroting data yang berhubungan dengan keselematan dan kesehatan kerja (K3).

## 2.4 Teori Program

### 2.4.1 CodeIgniter

Menurut Betha Sidik (2018:2) mengemukakan bahwa, "Codeigniter (CI) adalah framework pengembangan aplikasi (application development framework) dengan menggunakan PHP, suatu kerangka pembuatan program dengan menggunakan PHP". Pengembangan dapat langsung menghasilkan program dengan cepat, dengan mengikuti kerangka kerja untuk membuat yang telah disiapkan oleh framework CI ini.

Menurut Yuniar Supardi dan Ading Hermawan (2018:1) "*CodeIgniter* (CI) adalah aplikasi *open source* yang berupa *framework* dengan model MVC *(Model, View, Controller)* untuk membangun *website* dinamis dengan menggunakan PHP".




**Gambar 2.1** Alur *CodeIgniter*

File index.php berfungsi sebagai front controller. Menginisialisasi resource utama yang dibutuhkan untuk menjalankan *CodeIgniter*.

1. Router, memeriksa HTTP request untuk menentukan apa yang harus dilakukan.
2. Jika file cache ada, dikirim langsung ke browser, melewati eksekusi sistem normal.

### 2.4.2 Hypertext Markup Language (Html)

Menurut Abdul Kadir (2018:83) mengemukakan bahwa, "HTML (*HyperText Markup languange*) adalah kode dasar yang digunakan untuk

menyusun halaman web".

Menurut Henderson (2009:232) mengemukakan bahwa, "HTML *(Hyper Text Mark Up Language)* merupakan bahasa yang digunakan untuk mendeskripsikan struktur sebuah halaman web. HTML berfungsi untuk mempublikasi dokumen online. Statement dasar dari HTML disebut tags. Sebuah tag dinyatakan dalam sebuah kurung siku (<>). Tags yang ditujukan untuk sebuah dokumen atau bagian dari suatu dokumen haruslah dibuat berupa pasangan. Terdiri dari tag pembuka dan tag penutup. Dimana tag penutup menggunakan tambahan tanda garis miring (/) di awal nama tag".

### 2.4.3 Xampp

Menurut Riyanto (2010:1) mengungkapkan bahwa, "XAMPP merupakan paket PHP dan MySQL berbasis open source, yang dapat digunakan sebagai tool pembantu pengembangan aplikasi berbasis PHP".

Menurut Hidayatullah dan Kawistara (2017:125) dalam bukunya mengatakan bahwa, " *XAMPP support* untuk banyak sistem operasi seperti *Windows*, *Linux*, *Mac OS* dan *Solaris* sehingga tidak terdapat masalah ketika melakukan perpindahan sistem operasi".

### 2.4.4 Cascading Style Sheet (CSS)

Menurut Abdul Kadir (2018:143) mengemukakan bahwa, "CSS (*Cascading Style Sheet)* biasa digunakan pada dokumen web dan digunakan untuk mengatur tampilan elemen-elemen HTML pada layar. Kertas, dan bahkan media lain".

Menurut Henderson (2009:72) mengemukakan bahwa, "CSS kepanjangan dari Cascading Style Sheet adalah bahasa-bahasa yang merepresentasikan halaman web. Seperti warna, layout, dan font. Dengan menggunakan CSS, seorang web developer dapat membuat halaman web yang dapat beradaptasi

dengan berbagai macam ukuran layar. Pembuatan CSS biasanya terpisah dengan halaman HTML. Meskipun CSS dapat disisipkan di dalam halaman HTML. Hal ini ditujukan untuk memudahkan pengaturan halaman HTML yang memiliki rancangan yang sama".

### 2.4.5 MySQL

Menurut Abdul Kadir (2018:170) mengemukakan bahwa, "MYSQL merupakan sistem manajemen database terkenal yang sekarang dimiliki oleh Oracle dan salah satu produknya yang bernama MYSQL *Community Server* bersifat "*Open Source*".

Menurut Kurniawan (2010:16) mengemukakan bahwa, "MySQL adalah salah satu jenis database yang banyak digunakan untuk membuat aplikasi berbasis web yang dinamis. MySQL termasuk jenis RDBMS (*Relational Database Management Sistem*). MySQL ini mendukung Bahasa pemrograman PHP. MySQL juga mempunyai query atau bahasa SQL(*Structured Query Language*) yang simple dan menggunakan escape character yang sama dengan PHP".

### 2.4.6 Sublime Text

Menurut Faridl (2015:3), "*Sublime text* adalah teks editor berbasis *Python*, sebuah teks editor yang elegan, kaya fitur, *cross platform*, mudah dan simpel yang cukup terkenal dikalangan *developer* (pengembang), penulis dan desainer. Para programmer biasanya menggunakan sublime text untuk menyunting *source code* yang sedang ia kerjakan".

Menurut Bos (2014:12) menjelaskan, "Sublime Text merupakan salah satu text editor yang sangat powerful yang dapat meningkatkan produktivitas dan mengembangkan kualitas kode yang tinggi".

### 2.4.7 Android SDK

Menurut Nazruddin safaat (2011), "Android SDK adalah tools API (*Apllication Programming Interface)* yang diperlukan untuk mulai mengembangkan aplikasi pada platform Android menggunakan bahasa Java. SDK mencakup perangkat tools pengembangan yang komprehensif. Android SDK terdiri dari *debbuger*, *libraries*, handset emulator, dokumentasi".

### 2.4.8 Hypertext Preprocessor (PHP)

#### 2.4.8.1 Pengertian PHP

Menurut Abdul Kadir (2018:236) dalam bukunya mengatakan bahwa, "PHP merupakan sebuah bahasa pemrograman yang berjalan dalam sebuah web server (*server side*)".

Menurut Kurniawan (2010:2) mengemukakan bahwa, "PHP adalah bahasa pemrograman untuk dijalankan melalui halaman web, umumnya digunakan untuk mengolah informasi di internet. Sedangkan dalam pengertian lain PHP adalah singkatan dari Hypertext Preprocessor yaitu bahasa pemrograman webserver-side yang bersifat open source atau gratis. PHP merupakan script yang menyatu dengan HTML dan berada pada server".

#### 2.4.8.2 Pernyataan pada PHP

Menurut Abdul Kadir (2018:244), menjelaskan pernyataan pada PHP menyerupai kalimat dalam bahasa tulis manusia. Suatu pernyataan dapat berdiri sendiri, tetapi bisa juga tergantung oleh pernyataan lain. Contoh pernyataan yang tidak tergantung pernyataan lain adalah :

```
Print ("Tes…Tes…123!<br>");
```

Adapun dua pernyataan

```
While ($bil . "<br>");
```

```
$bil = $bil + 1 ;
```

Pada peryataan seperti

```
While ($bil . < 13)
{
    Print ($bil . "<br>");
    $bil = $bil + 1;
}
```

Tergantung oleh

```
While ($bil < 13)
```

Pernyataan yang tergantung oleh pernyataan lain biasa ditulis agak menjorok ke kanan terhadap pernyataan penentunya.

### 2.4.8.3 Dasar variabel pada PHP

Menurut Abdul Kadir (2018:245), menjelaskan Variabel merupakan suatu nama yang menyatakan wadah memori komputer yang di gunakan untuk menyimpan nilai dan nilainya bisa diubah ketika kode PHP dieksekusi. Dibawah ini merupakan scrip variabel PHP :

```
<!DOCTYPE HTML>
<HTML>
    <HEAD>
        <title>Variabel</title>
    </head>
    <body>
        <?php
            $nama = "Dian Pramana Putra";
```

```
                Print($nama . “<br>”;

                $nama = “Muhammad Tulus”;
                Print ($nama . “<br>”);

            ?>
        </body>
</html>
```