# BAB II

# TINJAUAN PUSTAKA

## II.1. Penelitian Terkait

Dalam penelitian terdahulu, penulis mengambil beberapa referensi yang berkaitan dengan latar belakang masalah yang penulis ambil. Salah satunya yaitu penelitian yang dilakukan oleh Ade Zulkarnain Hasibuan, Munjiat Setiani Asih, Herlina Harahap (2019), dengan judul "Penerapan *QR Code* dan Vigenere Chiper Dalam Sistem Pelaporan Juru Parkir Ilegal", yang berkisimpulan:

1. Dari Pengujian yang telah dilakukan dapat didapatkan juru parkir illegal di beberapa lokasi, hal ini diketahui dari tidak adanya kartu identitas yang dimiliki oleh juru parkir tersebut.
2. Dengan adanya sistem ini maka pemerintah dapat mengurangi aksi premanisme yang dilakukan oleh juru parkir.
3. Dengan adanya sistem pelaporan juru parkir ini diharapkan dapat meningkatkan pendapatan daerah melalui tarif parkir yang disetorkan oleh juru parkir legal.
4. Dengan adanya sistem ini dapat memudahkan masyarakat untuk melaporkan juru parkir illegal yang ada di beberapa tempat sehingga pemerintah dapat melakukan tindakan tegas terhadap juru parkir illegal tersebut.

Penelitian yang dilakukan oleh Moh. Lukman Sholeh, Luthfi Muharom (2016), dengan judul "*Smart Presensi* Menggunakan *Qr-Code* dengan enkripsi Vigenere Chiper", berkesimpulan implementasi pada integrasi Sistem Informasi

Akademik dapat menghubungkan data yang dikirim dari sistem absensi menggunakan *Qr-Code Scanner*.

Penelitian yang di lakukan oleh Akhmad Qashlim, Hasruddin (2015), dengan judul "Implementasi Teknologi *Qr-Code* untuk Kartu Indentitas" berkesimpulan *Qr-Code* merupakan gambar 2 dimensi yang dapat menyimpan indentitas pribadi, perusahaan dan URL *link* web untuk digunakan pada buku, majalah atau mengenkripsi data pribadi seseorang.

Penelitian yang dilakukan oleh Syaiful Bari, Ira Wahyu Mawaddah, dan Rachmansyah (2015) tentang "Aplikasi Validasi STNK Kenderaan Bermotor Menggunakan *Qr Code* Berbasis Android" berkesimpulan:

1. Dengan adanya aplikasi ini dapat memudahkan SAMSAT dan polisi LANTAS dalam mengidentifikasi keaslian STNK yang dimiliki, dengan demikian dapat membantu polisi mengungkap pemalsuan STNK dan pencurian kenderaan bermotor.
2. Menggunakan database berbasis *web* server membuat aplikasi ini dapat diakses dimanapun dan dapat diterapkan secara nasional dengan catatan masih terdapat jaringan internet di dalamnya.
3. Aplikasi ini efektif berjalan pada sistem operasi android berbasis 4.0 *Ice Cream Sandwich* ke atas, di buat menggunakan bahasa pemrograman java dan mengkonversi data menjadi *QR Code* serta menggunakan algoritma pada sistem pengamannya.

4. Aplikasi ini juga mudah untuk dijalankan dengan ukuran aplikasi yang kecil sehingga tidak memakan sumber daya yang besar pada *Smart Phone* yang berbasis android.

Penelitian yang dilakukan oleh Dadang Iskandar Mulyana (2016), yang berjudul "Kajian penerapan *Encode* Data Dengan Base64 Pada Pemrograman PHP" berkesimpulan penggunaan enkripsi dengan Base64 *Encode* sangat mudah dan dapat memungkinkan di terapkan dalam pahasa pemrograman PHP, baik penggunaan enkripsi dan *dekripsi* data, alamat URL ataupun gambar, satu hal yang menjadi cukup baik adalah jika data berupa gambar yang disimpan ke dalam database seperti halnya database *mysql* sangat efesien dalam proses *maintenance* data karna data yang disimpan berupa teks biasa bukan binary yang pada saat kompresi data *backup.*

Penelitian yang dilakukan oleh Khairul Fahmi (2015), yang berjudul "Sistem Penanganan tindak Pidana Pemilu (*System for The Crime of Election*)*"* berkesimpulan, sistem penanganan tindak pidana pemilu masih membutuhkan pembenahan agar dapat diterapkan dengan baik dan efektif untuk menjadi salah satu instrument mewujudkan pemilu yang jujur dan adil. Perbaikan sistem penanganan meliputi perbaikan regulasi, pengaturan kapasitas dan profesionalisme penegakan hukum pemilu dan peningkatan kesadaran hukum seluruh pemangku kepentingan pemilu. Tanpa melakukan itu, sistem penanganan tindak pidanan pemilu akan selalu sejalan di tempat dan tidak akan berhasil guna dalam menopang perwujudan pemilu yang jujur dan adil.

## II.2. Studi Literatur

### II.2.1. Bawaslu (Badan Pengawas Pemilu Pemilu)

Bawaslu (Badan Pengawas Pemilu) adalah Lembaga penyelenggara Pemilu yang bertugas mengawasi penyelenggaraan Pemilu di seluruh Indonesia, yang diatur dalam undang-undang 15 tahun 2011 tentang penyelenggara Pemilihan Umum.

Bawalu memiliki tugas, wewenang dan kewajiban berdasarkan Undang-Undang Nomor 15 Tahun 2017, yaitu:

1. Bawaslu menyusun standar tata laksana kerja pengawasan tahapan penyelenggaraan Pemilu sebagai pedoman kerja bagi pengawas Pemilu di setiap tingkatan.
2. Bawaslu bertugas mengawasi penyelenggaraan Pemilu dalam rangka pencegahan dan penindakan pelanggaran untuk terwujudnya Pemilu yang demokratis.
3. Dalam melaksanakan tugas, Bawaslu berwenang:
    a. Menerima laporan dugaan pelanggaran terhadap pelaksanaan ketentuan peraturan perundang-undangan mengenai Pemilu.
    b. Menerima laporan adanya dugaan pelanggaran administrasi Pemilu dan mengkaji laporan dan temuan, serta merekomendasikannya kepada yang berwenang.
    c. Menyelesaikan sengketa Pemilu.
    d. Membentuk Bawaslu Provinsi.
    e. Mengangkat dan memberhentikan anggota Bawaslu Provinsi.

f. Melaksanakan wewenang lain yang diatur dalam ketentuan peraturan perundang-undangan

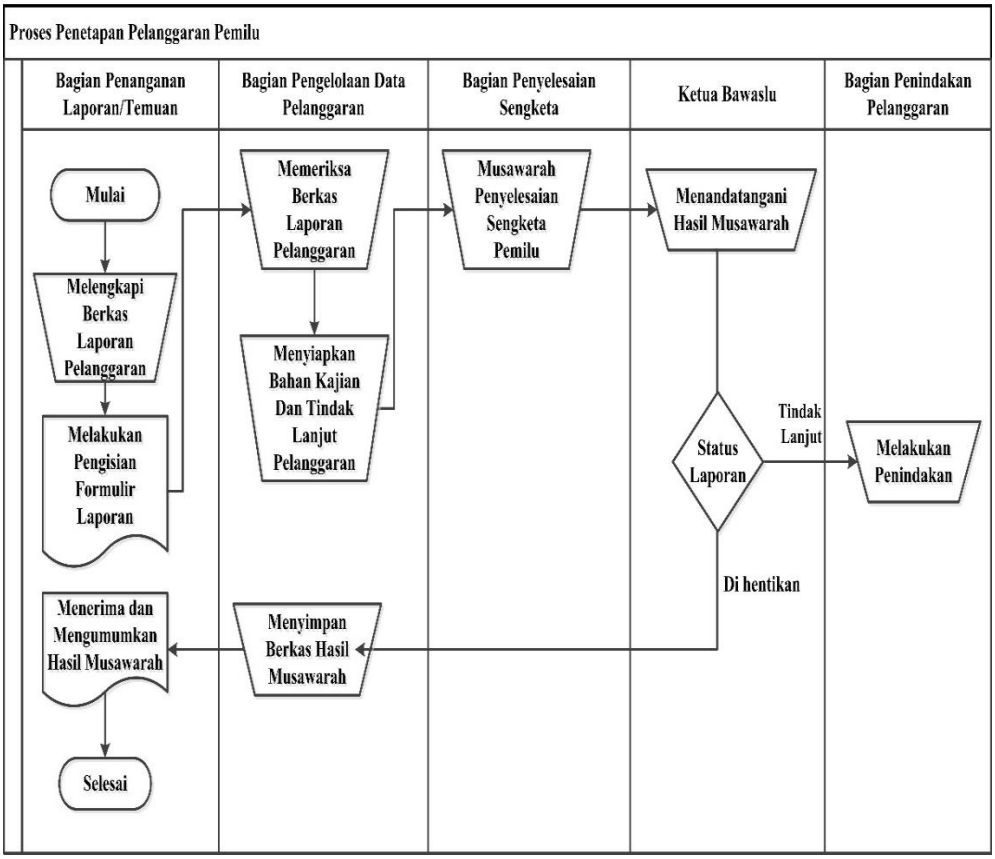
4. Bawaslu berkewajiban:
   a. Bersikap tidak diskriminatif dalam menjalankan tugas dan wewenangnya.
   b. Melakukan pembinaan dan pengawasan terhadap pelaksanaan tugas Pengawas Pemilu pada semua tingkatan.
   c. Menerima dan menindaklanjuti laporan yang berkaitan dengan dugaan adanya pelanggaran terhadap pelaksanaan peraturan perundang-undangan mengenai Pemilu.
   d. Menyampaikan laporan hasil pengawasan kepada Presiden, Dewan Perwakilan Rakyat, dan KPU sesuai dengan tahapan Pemilu secara periodik dan/atau berdasarkan kebutuhan.
   e. Melaksanakan kewajiban lain yang diberikan oleh peraturan perundang-undangan.

### II.2.2. Validasi

Validasi adalah suatu tindakan pembuktian dengan cara yang sesuai bahwa tiap bahan, proses, prosedur, kegiatan, sistem, perlengkapan atau mekanisme yang digunakan dalam produksi dan pengawasan akan senantiasa mencapai hasil yang diinginkan.

### II.2.3. Prosedur Penanganan Pelaporan Pelanggaran Pemilu

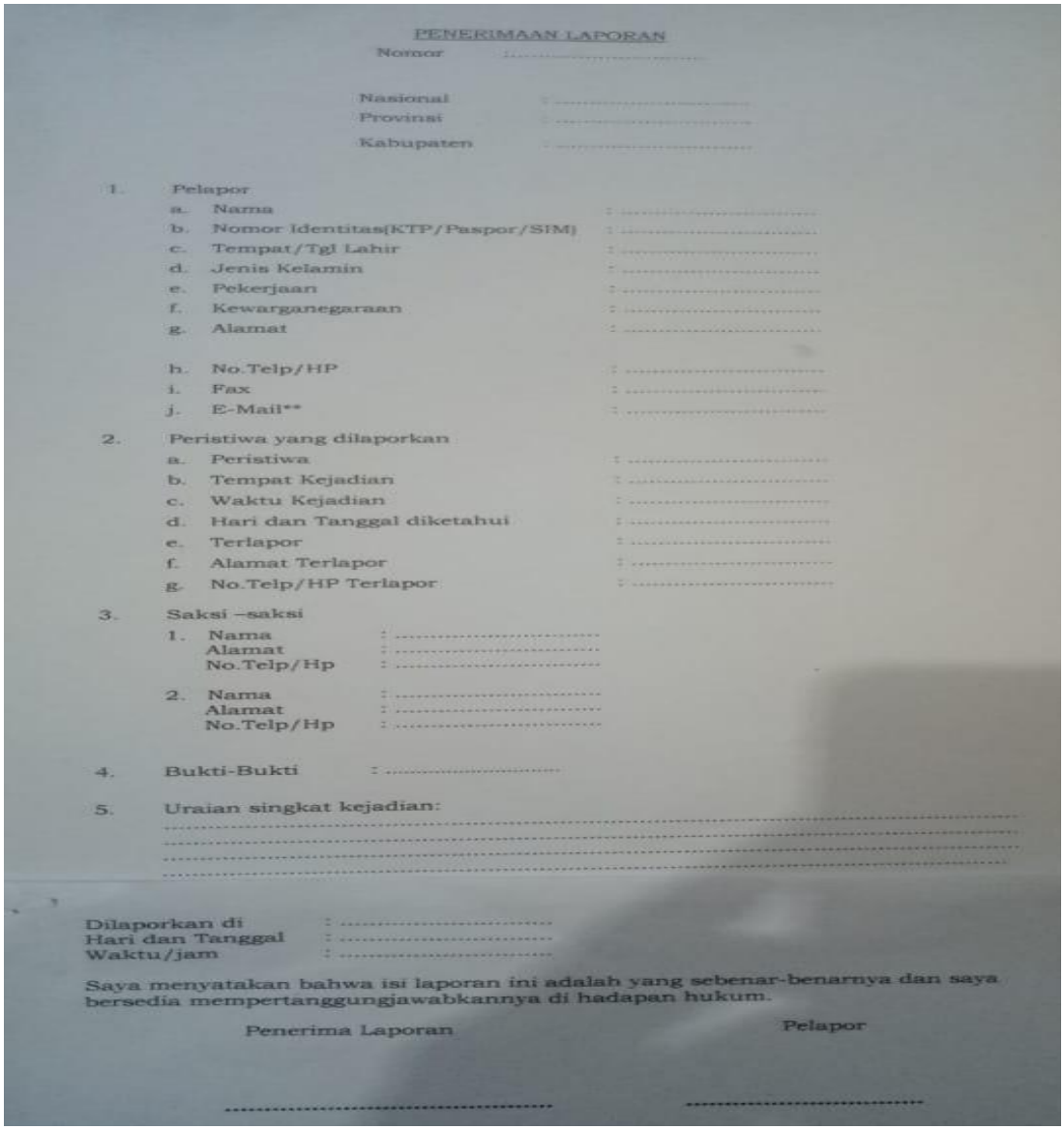
Prosedur penanganan pelaporan pelanggaran Pemilu di Bawaslu Sumatera Utara dapat dilihat dalam bentuk *Flow of Document* (FOD) pada gambar II.1.




**Gambar II.1. FOD (*Flow OF Document*) Prosedur Penanganan Pelaporan Pelanggaran Pemilu Di Bawaslu Sumatera Utara**
*(Sumber : Bawaslu Sumatera Utara:2019)*

### II.2.4. Formulir Pelanggaran Pemilu

Formulir Pelanggaran Pemilu adalah lembaran kartu/kertas dengan ukuran tertentu yang di dalamnya terdapat data/informasi tentang laporan pelanggaran Pemilu. Berikut contoh formulir pelanggaran pemilu dapat dilihat pada Gambar II.2.



PENERIMAAN LAPORAN

Nomor : .....

Nasional : .....
Provinsi : .....
Kabupaten : .....

1. Pelapor
   a. Nama : .....
   b. Nomor Identitas(KTP/Paspor/SIM) : .....
   c. Tempat/Tgl Lahir : .....
   d. Jenis Kelamin : .....
   e. Pekerjaan : .....
   f. Kewarganegaraan : .....
   g. Alamat : .....
   h. No.Telp/HP : .....
   i. Fax : .....
   j. E-Mail** : .....
2. Peristiwa yang dilaporkan
   a. Peristiwa : .....
   b. Tempat Kejadian : .....
   c. Waktu Kejadian : .....
   d. Hari dan Tanggal diketahui : .....
   e. Terlapor : .....
   f. Alamat Terlapor : .....
   g. No.Telp/HP Terlapor : .....
3. Saksi –saksi
   1. Nama : .....
      Alamat : .....
      No.Telp/Hp : .....
   2. Nama : .....
      Alamat : .....
      No.Telp/Hp : .....
4. Bukti-Bukti : .....
5. Uraian singkat kejadian:
   .....
   .....
   .....
   .....

Dilaporkan di : .....
Hari dan Tanggal : .....
Waktu/jam : .....

Saya menyatakan bahwa isi laporan ini adalah yang sebenar-benarnya dan saya bersedia mempertanggungjawabkannya di hadapan hukum.

Penerima Laporan Pelapor

..... .....

**Gambar II.2. Formulir Laporan Pelanggaran Pemilu**
(*Sumber: Bawaslu Sumatera Utara:2019*)

### II.2.5. *Qr-Code*

*Qr-Code* (*Quick Response Code*) adalah *barcode* dua dimensi yang dapat menyimpan data. Qr-Code di kembangkan oleh Denso Corporation, Jepang dan dapat digunakan secara gratis, bahkan untuk keperluan komersial. *Qr-Code* dipublikasikan tahun 1994 dengan fungsional utama yaitu dapat dengan mudah dibaca oleh pemindai *Qr-Code*. Tujuan dari *Qr-Code* ini adalah untuk

menyampaikan informasi dengan cepat dan mendapatkan respon yang cepat. *Qr-Code* dapat menyimpan informasi secara horizontal dan vertikal. Berikut contoh gambar *Qr-Code* dapat dilihat pada Gambar II.3.

**Gambar II.3. *Qr-Code* (*Quick Response Code*)**
(*Sumber: Moh. Lukman Sholeh, dkk:2016*)

### II.2.6. Base64

Base64 adalah salah satu algoritma untuk *encoding* dan *decoding* suatu data ke dalam format *ASCII*, yang didasarkan pada pola bilangan 64 atau bisa dikatakan sebagai salah satu metoda yang digunakan untuk melakukan *encoding* (penyandian) terhadap data *binary*. Karakter yang dihasilkan pada transformasi Base64 ini terdiri dari A sampai Z, a sampai z dan 0 sampai 9, serta ditambah simbol "+" dan "/" serta satu buah karakter sama dengan (=) di karakter terahir yang di pakai untuk pengisian *pad* atau dengan penyesuaian menggenapkan *binary*. Karakter simbol yang akan dihasilkan akan tergantung dari proses algoritma yang berjalan.

Teknik *encoding* Base64, jika satu senarai (*string*) *bytes* yang akan disandikan dengan Base64 maka caranya adalah:

a. Masukkan kata yang akan di *encoding* dengan Base64, contoh "Ali Aman"
b. Ambil nilai *binary* dari *ASCII* setiap karakter yaitu:

   A=65, *binary*=01000001

   l= 108, *binary*=01101100

i= 105, *binary*=01101001

= 32, *binary*=00100000

A=65, *binary*=01000001

m=109, *binary*=01101101

a=97, *binary*=01100001

n=110, *binary*=01101110

c. Deretan *binary* dari “Ali Aman”: 01000001 01101100 01101001 00100000 01000001 01101101 01100001 01101110.

d. Deretan *binary* (dijadikan 1 baris panjang) dari "Ali Aman": 0100000101101100011010010010000001000001011011010110000101101110

e. Proses per 6 bit, setiap 6 bit akan mewakili 1 karakter Base64:

0100000101101100011010010010000001000001011011010110000101101110.

Maka menjadi 010000-010110-110001-101001-001000-000100-000101-101101-011000-010110-1110.

f. Panjang kata bukan kelipatan 3, akan *padding* “=” sepanjang 1 karakter.

g. Maka hasil dari *encoding* nya adalah:

| *Binary* | Desimal | Base64 |
|---|---|---|
| 00010000 | 16 | Q |
| 00010110 | 22 | W |
| 00110001 | 49 | x |
| 00101001 | 41 | p |
| 00001000 | 8 | I |
| 00000110 | 6 | G |
| 00000101 | 5 | F |
| 00101101 | 45 | t |
| 00011000 | 24 | Y |

| | | |
|---|---|---|
| 00010110 | 22 | W |
| 00111000 | 56 | 4 |

h. Hasil *encoding* Base64, dengan *padding: "QWxpIGFtYW4="*.

### II.2.7. *Website*

*Website* adalah suatu halaman web yang saling berhubungan yang umumnya berisikan kumpulan informasi berupa data teks, gambar, animasi, audio, video maupun gabungan dari semuanya yang biasanya dibuat untuk personal, organisasi dan perusahaan. *Website* dibedakan menjadi dua yaitu bersifat statis dan dinamis. Bersifat statis apabila isi informasinya tetap dan isi informasinya hanya dari pemilik *website*, sedangkan *website* bersifat dinamis apabila isi informasinya selalu berubah-ubah dan dapat diubah oleh pemilik maupun pengguna *website*. Contoh *website* statis adalah *website profile* perusahaan, sedangkan contoh *website* dinamis seperti *facebook, twiter* dan lain-lain.

### II.2.8. HTML (*Hypertext Markup Language)*

HTML (*Hypertext Markup Language)* adalah sebuah markah yang digunakan untuk membuat sebuah halaman *website.* Didalam dunia pemrograman berbasis *website HTML* menjadi pondasi dasar pada halaman *website.* Sebuah file HTML disimpan dengan ekstensi .html (*dot* html) dan dapat di akses menggunakan *web browser,* seperti *Mozilla Firefox, Opera, Google Chrome* dan lain-lain. Untuk membuat *website* tidak cukup dengan hanya dengan menggunakan HTML, untuk pembuatannya memerlukan CSS, *JavaScript,* dan PHP. HTML memiliki beberapa elemen yang tersusun dari tag-tag yang memiliki fungsi masing-masing, seperti tag *heading, paragraph,* pembuatan *form,* tombol, *list,* membuat *hyperlink,* yang menggabungkan antar halaman *website*.

### II.2.9. CSS (*Cascading Style Sheets*)

CSS merupakan singkatan dari *Cascading Style Sheets*. Sesuai dengan namanya CSS memiliki sifat *style sheet language* yang berarti bahasa pemrograman yang digunakan untuk *web design.*

### II.2.10. *JavaScript*

*JavaScript* adalah bahasa pemrograman *website* yang bersifat *Client Side Programing Language*. *Client Side Programing Language* adalah tipe bahasa pemrograman yang prosesnya dilakukan oleh *client.* Bahasa pemrograman *Client Side* berbeda dengan bahasa pemrograman *Server Side* seperti PHP, dimana untuk *server side* seluruh kode program dijalankan di sisi server.

### II.2.11. PHP

PHP adalah bahasa pemrograman *script server-side* yang di desain untuk pengembangan *website.* PHP dikembangkan pada tahun 1995 oleh Rasmus Lerdorf, dan sekarang dikelola oleh *The PHP Group.*

PHP disebut pemrograman *sever side* karena PHP di proses pada komputer server. Pada awalnya PHP merupakan singkatan dari *Personal Home Page.* Saat ini PHP singkatan dari *Hypertext Preprocessor* sebuah kepanjangan *rekursif*, yakni permainan permainan kata dimana kepanjangan terdiri dari singkatan itu sendiri.

### II.2.12. MySQL

MySQL adalah sebuah *database management system* (manajemen basis data) menggunakan perintah dasar SQL (*Structured Query Language*). MySQL masuk kedalam jenis RDBMS (*Relational Database Management System*). SQL sendiri merupakan suatu bahasa yang dipakai dalam pengambilan data pada

*relational database* atau database terstruktur. Jadi MySQL adalah *database management system* yang menggunakan bahasa SQL sebagai bahasa penghubung antara perangkat lunak aplikasi dengan database server.

### II.2.13. XAMPP

XAMPP adalah perangkat lunak bebas yang mendukung banyak sistem operasi, merupakan kompilasi dari beberapa program. XAMPP merupakan *tool* yang menyediakan paket perangkat lunak ke dalam satu buah paket. Dengan menginstall XAMPP maka tidak perlu lagi melakukan instalasi dan konfigurasi *web server* Apache, PHP dan MySQL secara manual. XAMPP akan mengistalasi dan mengkonfigurasi secara otomatis.

### II.2.14. Sublime Text

Sublime Text Editor adalah editor teks untuk berbagai bahasa pemograman termasuk pemograman PHP. *Sublime Text Editor* merupakan *editor text* lintas *platform* dengan Python *application programming interface* (API). *Sublime Text Editor* juga mendukung banyak bahasa pemrograman dan bahasa *markup,* dan fungsinya dapat ditambah dengan *plugin*.

### II.2.15. *Unified Modeling Language* (UML)

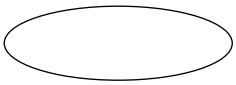
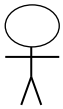
*Unified Modelling Language* (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasi sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem.

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

1. *Use Case* Diagram

*Use case* diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram dapat dilihat pada tabel II.1.

**Tabel II.1. Simbol *Use Case***

| Gambar | Keterangan |
|---|---|
|  | *Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukar pesan antar unit dengan aktor, dan dinyatakan dengan menggunakan kata kerja di awal nama *use case*. |
|  | Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki *control* terhadap *use case*. |
|  | Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data. |
|  | Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem. |

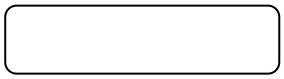
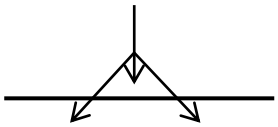
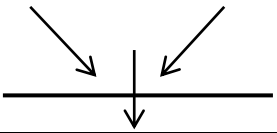
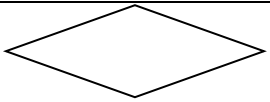
| | |
|---|---|
|  | *Include*, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program. |
|  | *Extend*, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi. |

**(Sumber : Urva dan Siregar, 2015 : 94)**

2. Diagram Aktivitas (*Activity Diagram*)

*Activity Diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram* dapat dilihat pada tabel II.2.

**Tabel II.2. Simbol *Activity Diagram***

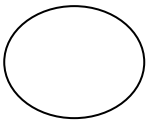
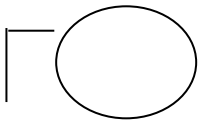
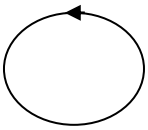
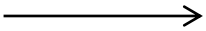
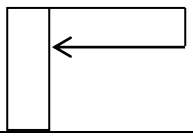
| Gambar | Keterangan |
|---|---|
|  | *Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas. |
|  | *End point*, akhir aktifitas. |
|  | *Activites*, menggambarkan suatu proses/kegiatan bisnis. |
|  | *Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu. |
|  | *Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi. |
|  | *Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false*. |
|   | *Swimlane*, untuk menunjukkan siapa melakukan apa. |

**(Sumber : Urva dan Siregar, 2015 : 94)**

3. Diagram Urutan (*Sequence Diagram*)

*Sequence diagram* menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram* dapat dilihat pada tabel II.3.

**Tabel II.3. Simbol *Sequence Diagram***

| Gambar | Keterangan |
|---|---|
|  | *Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data. |
|  | *Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak. |
|  | *Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek. |
|  | *Message*, simbol mengirim pesan antar *class*. |
|  | *Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri. |
|  | *Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi. |
|  | *Lifeline*, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*. |

**(Sumber : Urva dan Siregar, 2015 : 95)**

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggung jawab entitas yang menentukan perilaku sistem. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti yang dapat dilihat pada tabel II.4.

**Tabel II.4. Simbol *Multiplicity Class Diagram***

| Multiplicity | Penjelasan |
|---|---|
| 1 | Satu dan hanya satu |
| 0..* | Boleh tidak ada atau 1 atau lebih |
| 1..* | 1 atau lebih |
| 0..1 | Boleh tidak ada, maksimal 1 |
| n..n | Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4 |

**(Sumber : Urva dan Siregar, 2015 : 95**