BAB 2
LANDASAN TEORI

2.1 *Automatic License Plate Recognition (ALPR)*

Pengenal Plat Lisensi Otomatis *(Automatic License Plate Recognition (ALPR))* atau juga dikenal sebagai sistem Pengenal Pelat Nomor Otomatis *(Automatic Numbert Plate Recognition (ANPR))* adalah solusi teknologi untuk menangkap gambar kendaraan dan kemudian menyempurnakannya menggunakan berbagai teknik Pemrosesan Gambar [3].

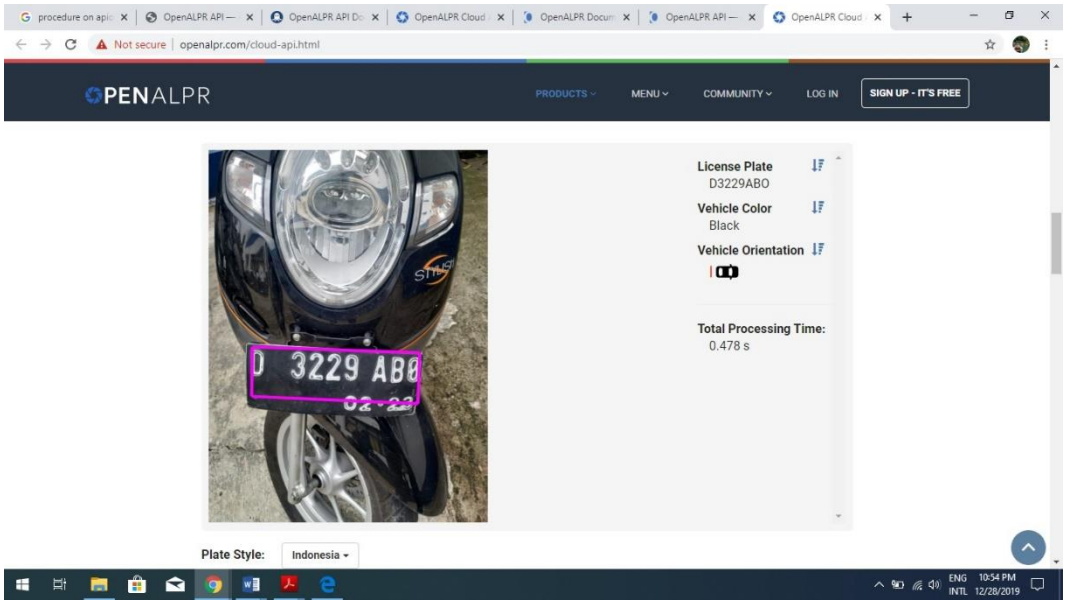
Langkah pengenalan plat kendaraan pada sistem ALPR secara umum terdiri atas 4 (empat) tahapan penting dalam yaitu deteksi plat nomor, segmentasi karakter, data ekstrasi data, dan pengenalan karakter. Beberapa penelitian mengenai ALPR telah dilakukan dengan menawarkan beberapa metode. Metode saat ini yang banyak digunakan untuk deteksi plat nomor, yaitu metode berbasis *image processing* dan metode berbasis *machine learning* [4].

Gambar 2. 1 Langkah-langkah proses dalam ALPR

2.2 API *Open* ALPR

API openALPR adalah sebuah aplikasi open souce berbasis web yang melayani analisis gambar untuk pelat nomor serta informasi kendaraan seperti merek, model, dan warna. Layanan ini menggunakan Cloud API yang diintegrasikan ke dalam sebuah aplikasi melalui layanan REST berbasis web.

Melalui layanan di aplikasi API openALPR ini, kita tinggal mengirimkan gambar kendaraan. Dan selanjutnya, seluruh proses mengubah gambar pelat kendaraan ke dalam bentuk teks akan diproses oleh aplikasi tersebut.



Gambar 2. 2 Contoh hasil pengolahan gambar plat kendaraan

2.3 REST *Web Service*

REST Client API adalah antarmuka RESTful untuk membangun aplikasi klien. Pada arsitektur RESTful, REST server menyediakan *resources* (sumber daya/data) dan REST client mengakses dan menampilkan *resource* tersebut untuk penggunaan selanjutnya. Kemampuan API meliputi yang berikut:

- Membuat, mengambil, memperbarui, dan menghapus dokumen, metadata, dan tiga kali lipat semantik dalam *database*.
- Cari dokumen dan grafik semantik dan nilai leksikon *query*, menggunakan beberapa format *query*, termasuk *query* string, *query* terstruktur, *query* gabungan.
- Penyesuaian *query* dengan mengkonfigurasi opsi *query* yang dinamis.
- Menerapkan transformasi untuk mendokumentasikan konten dan hasil pencarian.
- Memerluas API untuk mengekspos kemampuan khusus yang Anda buat di XQuery dan instal di sebuah Server.

Rest Client API bekerja dengan XML, JSON, teks, dan dokumen biner. Dalam kebanyakan kasus, sebuah aplikasi dapat menggunakan XML atau JSON untuk bertukar data non-dokumen seperti querry. Contohnya, aplikasi Rest Client API berinteraksi dengan sebuah server melalui instance REST API, HTTP App Server yang dikonfigurasi secara khusus. Setiap *instance* REST API dimaksudkan untuk melayani basis data konten tunggal dan aplikasi klien. Anda dapat membuat dan mengonfigurasi instance REST API melalui permintaan REST atau secara interaktif [5].

(*Sumber :* MarkLogic Server, 2019, *Rest Application Developer's Guide*, MarkLogic 10, MarkLogic Corporation)

Gambar 2. 3 Arsitektur REST Web Service

2.4 Cpanel

Cpanel adalah suatu kontrol panel dari web hosting memiliki fungsi untuk mengelola pengaturan domain, hosting ataupun website. Menurut Wikipedia, cPanel akan memberikan tampilan grafis dan automasi untuk memudahkan proses hosting di sebuah situs web. cPanel dapat berjalan dalam server/ hosting dengan sistem operasi centos, Red Hat Linux, dan FreeBSD. Cpanel biasanya dipasang pada Dedicated Server atau Virtual Private Server (VPS) yang menggunakan sistem operasi Linux, FreeBSD, dan sejenisnya. Aplikasi-aplikasi yang didukung cPanel meliputi Apache, PHP, mySQL,Postgres, Perl, Python, and BIND, dengan email seperti POP3, IMAP, layanan-layanan SMTP [6].

2.4.1 MySQL

MySQL merupakan *database engine* atau *server database* yang mendukung Bahasa database pencarian sql. Mysql adalah sebuah perangkat lunak system manajemen basis data sql atau dbms yang multithread dan multi user. Mysql ab membuat mysql tersedia sebagai perangkat lunak gratis dibawah lisensi GNU General Public License (GPL). Mereka juga menjual dibawah lisensi komersial untuk kasu-kasus dimana penggunaannya tidak cocok untuk penggunaan GPL.

Mysql merupakan turunan salah satu konsep utama dalam database sejak lama, yaitu sql (structured query language). Sql adalah sebuah konsep pengoperasian database, terutama untuk pemilihan atau seleksi dan pemasukkan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah dan cepat secara otomatis. Keandalannya suatu system database (DBMS) dapat diketahui dari cara kerja optimizernya dalam melakukan perintah-perintah sql, yang dibuat oleh user maupun program-program aplikasinya. Sebagai database server, mysql dapat dikatakan lebih unggul dibandingkan database server lainnya dalam hal query data. Hal ini terbukti untuk query yang dilakukan oleh single user, kecepatan query mysql bisa sepuluh kali lebih cepat dari postgresql dan lima kali lebih cepat dibandingkan interbase [7].

Kepopuleran MySQL antara lain karena MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses databasenya sehingga mudah untuk digunakan, kinerja query cepat, dan mencukupi untuk kebutuhan database perusahaan-perusahaan yang berskala kecil sampai menengah, MySQL juga bersifat open source (tidak berbayar). MySQL merupakan database yang pertama kali didukung oleh bahasa pemrograman script untuk internet (PHP dan Perl). MySQL dan PHP dianggap sebagai pasangan software pembangun aplikasi web yang ideal. MySQL lebih sering digunakan untuk membangun aplikasi berbasis web, umumnya pengembangan aplikasinya menggunakan bahasa pemrograman script PHP.

2.4.2 PhpMyAdmin

phpMyAdmin adalah aplikasi web untuk mengelola database MySQL dan database MariaDB dengan lebih mudah melalui antarmuka (interface)grafis.

Aplikasi web ini ditulis menggunakan bahasa pemrograman PHP. Sebagaimana aplikasi-aplikasi lain untuk lingkungan web (aplikasi yang dibuka atau dijalankan menggunakan browser), phpMyAdmin juga mengandung unsur HTML/XHTML, CSS dan juga kode JavaScript. Aplikasi web ini ditujukan untuk memudahkan pengelolaan basis data MySQL dan MariaDB dengan penyajian antarmuka web yang lengkap dan menarik [8].

phpMyAdmin merupakan aplikasi web yang bersifat open source (sumber terbuka) sejak pertama dibuat dan dikembangkan. Dengan dukungan dari banyak developer dan translator, aplikasi web phpMyAdmin mengalami perkembangan yang cukup pesat dengan ketersediaan banyak pilihan bahasa. Sampai saat ini, ada kurang lebih 65 bahasa yang sudah didukung oleh aplikasi web phpMyAdmin.

Keberadaan phpMyAdmin yang dianggap penting dan sifatnya yang merupakan sumber terbuka menjadikannya salah satu aplikasi yang selalu ada di cPanel (aplikasi populer untuk pengontrol website). Hal ini menunjukkan bahwa penyedia web hosting (web hosting provider) menaruh kepercayaan yang sangat bersar pada phpMyAdmin sebagai salah satu aplikasi web yang dipasang (install) di server.

2.5 *JavaScript Object Notation* (JSON)

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh komputer. Format ini dibuat berdasarkan bagian dari Bahasa Pemprograman JavaScript, Standar ECMA-262 Edisi ke-3 - Desember 1999. JSON merupakan format teks yang tidak bergantung pada bahasa pemprograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran-data [9].

JSON terbuat dari dua struktur:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar berkunci (*keyed list*), atau associative array.

2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemprograman moderen mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemprograman yang juga berdasarkan pada struktur data ini [9].

2.6 HTML

Hyper Text Markup Language (HTML) adalah sebuah bahasa markah yang digunakan untuk membuat sebuah halaman web. Weibsite yang dibuat dengan HTML ini, dapat di lihat oleh semua orang yang terkoneksi dengan internet . Tentunya dengan menggunakan aplikasi penjelajah atau browser [10].

Tiap kali kita mengakses dokumen web, maka sesungguhnya kita mengakses dokumen seseorang yang ditulis dengan menggunakan format HTML. Beberapa orang merasa keberatan jika dikatakan HTML adalah sebuah bahasa pemrograman karena struktur yang dimilikinya dianggap terlalu sederhana, kode-kode dibaca oleh browser baris per baris, dari atas ke bawah. HTML juga tidak memiliki ‘looping’ seperti bahasa pemrograman lain.

Pada HTML dipergunakan hypertext link atau hubungan antara teks dan dokumen lain. Dengan demikian pembaca dokumen bisa melompat dari satu dokumen ke dokumen yang lain dengan mudah.

2.7 PHP

Menurut Betha Sidik (2017), PHP secara umum dikenal sebagai bahasa pemrograman *script-script* yang membuat dokumen HTML secara in the fly yang di eksekusi di *server web,* dokumen HTML yang dihasilkan dari suatu aplikasi bukan dokumen HTML yang dibuat dengan menggunakan editor teks atau editor HTML. Dikenal juga sebagai bahasa pemrograman *server side*. Dengan menggunakan PHP maka *maintenance* suatu situs *web* menjadi lebih mudah. Proses

update data dapat dilakukan dengan menggunakan aplikasi yang dibuat dengan menggunakan script PHP [11].

PHP/FI merupakan nama awal dari PHP. PHP – Personal Home Page, FI adalah *Form Interface*. Dibuat pertama kali oleh *Rasmus Lerdoff*. PHP, awalnya merupakan program CGI yang dikhususkan untuk menerima input melalui *form* yang ditampilkan dalam browser *web*. *Software* ini disebarkan dan dilisensikan sebagai perangkat lunak *Open Source*. CGI (*Common Gateway Interface*) sendiri menurut Beta Sidik adalah suatu standar yang menghubungkan (*interface*) aplikasi eksternal dengan *server web*.

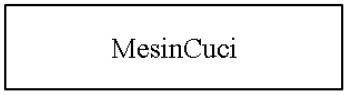
Kini, PHP adalah kependekan dari PHP:*HyperText Preprocessor*(rekursif mengikut gaya penamaan di *nix), merupakan bahasa utama *script server side* yang disisipkan pada HTML yang dijalankan di *server*, dan juga bisa digunakan untuk

2.8 Pemodelan dan Uml

UML(*Unified Modelling Language*) merupakan kesatuan dari bahasa pemodelan yang dikembang oleh Booch, *Object Modelling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE).metode *Booch* dari *Grady Booch* sangat terkenal dengan nama metode *Design Object Oriented.* Metode ini menjadikan proses analisis dan desain ke dalam empat tahapan iteratif, yaitu:identifikasi kelas-kelas dan obyek-obyek, identifikasi semantik dari hubungan obyek dan kelas tersebut, pencarian *interface* dan implementasi. Keunggulan metode *Booch* adalah pada detail dan kayanya dengan notasi dan elemen [12].

2.9 *Class* Diagram

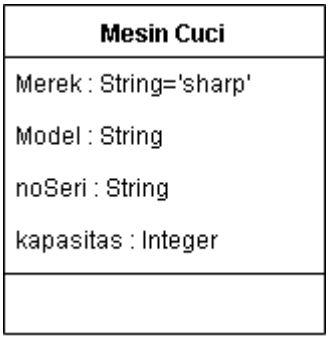
Class, dalam notasi UML digambarkan dengan kotak. Nama *class* menggunakan huruf besar di awal kalimatnya. Dan diletakkan di atas kotak. Bila *class* mempunyai nama terdiri dari 2 suku kata atau lebih, maka semua suku kata digabungkan tanpa spasi dengan huruf awal tiap suku kata menggunakan huruf besar [12].



(Sumber : Pemodelan Visual dengan UML [Munawar])

Gambar 2. 4 Notasi *Class*

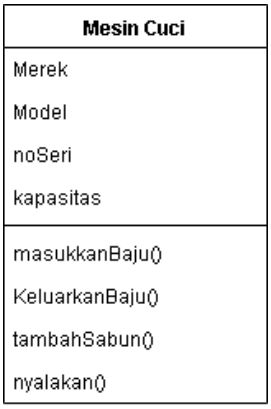
1. *Attribute*
   Attribute adalah properti dari sebuah *class*. *Attribute* ini melukiskan batas nilai yang mungkin ada pada obyek dari *class*. Sebuah *class* mungkin mempunyai nol atau lebih *attribute*. Secara konvensi, jika nama *attribute* terdiri atas satu suku kata, maka ditulis dengan huruf kecil. Akan tetapi jika nama *attribute* mengandung lebih dari satu suku kata maka semua suku kata digabungkan dengan suku kata pertama menggunakan huruf kecil dan awal suku kata berikutnya menggunakan huruf besar [12].



(Sumber : Pemodelan Visual dengan UML (Munawar))

Gambar 2. 5 *Attribute*

2. *Operation*
   Operation adalah suatu yang bisa dilakukan oleh sebuah *class* atau yang anda (atau *class* yang lain) dapat dilakukan untuk sebuah *class.* Seperti halnya *attribute*, nama *operation* juga menggunakan huruf kecil semua jika terdiri dari satu suku kata. Akan tetapi jika lebih dari satu suku kata, maka semua suku kata digabungkan dengan suku kata pertama huruf kecil dan huruf awal tiap suku berikutnya dengan huruf besar [12].



(Sumber : Pemodelan Visual dengan UML [Munawar])

Gambar 2. 6 *Operation*

Berikut adalah tabel operator yang dipakai pada suatu class diagram:

Tabel 2. 1 Operator

Operator	Visibility	Keterangan
+	*Public*	Sebuah *operation* atau *attribute* bisa di akses oleh siapapun
-	*Private*	Fitur yang hanya digunakan oleh *instance* dari *class*
#	*Protected*	Seperti private tetapi boleh di akses oleh anak-anaknya
-	*Package*	Bisa diakses langsung oleh *instance* dengan *package* sama

2.10 *Use Case* Diagram

Use case adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara user (pengguna) sebuah sistem dengan sistemnya sendiri melalu sebuah cerita bagaiamana sebuah sistem dipakai. Urutan langkah-langkah yang menerangkan antara pengguna dan sistem disebut skenario. Setiap skenario mendeskripsikan urutan kejadian. Setiap urutan diinisialisasi oleh orang, sistem yang lain, perangkat keras atau urutan waktu. Dengan demikian secara singkat bisa dikatakan *use case* adalah serangkaian skenario yang digabungkan bersama-sama oleh tujuan umum pengguna [13].

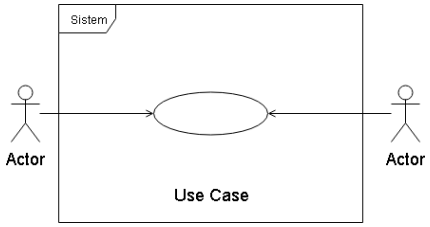
Dalam use case, pengguna biasanya disebut dengan aktor, aktor adalah sebuah peran yang bisa dimainkan oleh pengguna dalam interaksinya dengan sistem.

Model *use case* adalah bagian dari mode *requirement*(*Jacobson et all*,1992). Termasuk disini adalah *problem domain object model* dan penjelasan tentang user

interface. *Use case* memberikan spesifikasi fungsi-fungsi yang di tawarkan oleh sistem dari perspektif user.

1. Notasi *Use Case*

   Diagram *use case* menunjukkan 3 aspek dari sistem yaitu *actor, use case*, dan sistem/sub sistem boundari. *Actor* mewakili peran orang, sistem yang lain atau alat ketika berkomunikasi dengan *use case*. Gambar 2.21 mengilustrasikan *actor, use case* dan boundari [12].



(Sumber : Pemodelan Visual dengan UML [Munawar])

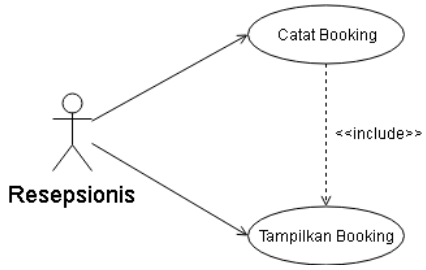
Gambar 2. 7 *Use Case* Model

2. Identifikasi *Actor*

   Dalam mengidentifikasi *actor*, terlebih dahulu menentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. *Actor* adalah *abstraction* dari orang dan sistem yang lain yang mengaktifkan fungsi dari target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa *actor* berinteraksi dengan *use case*, tetapi tidak memiliki kontrol atas *use case* [12].

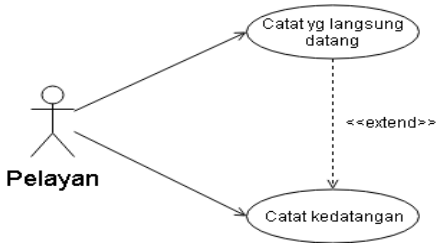
3. *Stereotype*

   Stereotype adalah sebuah model khusus yang terbatas untuk kondisi tertentu. Untuk menunjukkan stereotype digunakan simbol "<<" di awalnya dan ditutup ">>" di akhirnya. <<*extend*>> digunakan untuk menunjukkan bahwa satu *use case* merupakan tambahan fungsional dari *use case* yang lain jika kondisi atau syarat tertentu yang dipenuhi. Sedangkan <<*include*>> digunakan untuk menggambarkan bahwa suatu *use case* seluruhnya merupakan fungsionalitas dari use case lainnya. Biasanya <<*include*>> digunakan untuk menghindari pengkopian suatu *use case* karena sering dipakai [12].



(Sumber : Pemodelan Visual dengan UML [Munawar])

Gambar 2. 8 *Use Case Inclusion*

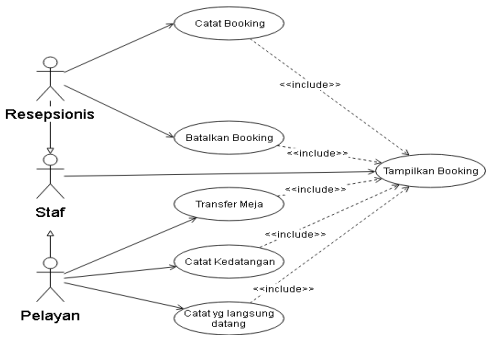


(Sumber : Pemodelan Visual dengan UML [Munawar])

Gambar 2. 9 *Use Case Extension*

4. Level *Use Case*

   Masalah umum dengan *use case* adalah dengan memfokuskan pada interaksi diantara dua user dan sistem. Sering kali perubahan pada proses bisnis adalah jalan terbaik untuk menyelesaikan masalah. Oleh karena itu *use case* perlu dibedakan menjadi sistem *use case* dan *business* use case. Sistem *use case* adalah sebut interaksi dengan *software*, sedangkan *businees use case* lebih menekankan kepada bagaimana sebuah bisnis merespon ke pelanggan atau kejadian/*event.* Berikut contoh pada gambar 2.27.



(Sumber : Pemodelan Visual dengan UML [Munawar])

Gambar 2. 10 *Use Case Extension*

5. Skenario *Use case* / Dokumentasi *Use Case*

   Setiap *use case* harus dideskripsikan dalam dokumen yang disebut dengan dokumen *flow of event.* Dokumen/skenario ini mendefinisikan apa yang harus dilakukan oleh sistem ketika aktor mengaktifkan *use case*. Struktur dokumen/skenario *use case* ini bisa bermacam-macam, tetapi umumnya deksripsi ini paling tidak harus mengandung [12]:

   1) *Brief Description* yaitu deskripsi singkat.
   2) *Actor* yang terlibat
   3) *Precondition* yang penting bagi use case untuk memulai
   4) *Main flow* dari kejadian yang bisa dirinci lagi menjadi *sub flow* yang lebih kecil agar dokumen lebih mudah di baca dan dimengerti.
   5) *Alternatif flow* untuk mendefinisikan situasi perkecualian
   6) *Postcondition* yang menjelaskan *state* dari sistem setelah *use case* berakhir.

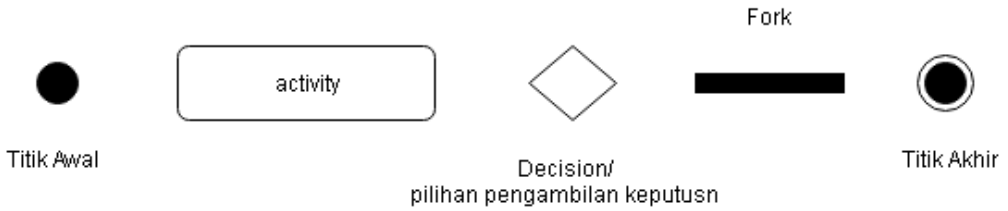
Tabel 2. 2 Contoh Dokumen/Skenario *use case*

Use Case	Nama use case
Biref Description	Deskripsi singkat
Actor	Aktor yang terlibat
Precondition	Yang penting bagi use case untuk memulai
Main flow	Kejadian yang bisa di rinci
Alternatif flow	Mendefinisikan situasi alternatifnya/pengecualian
Postcondition	Menjelaskan hasil dari *state* yang sudah dilakukan

2.11 *Activity* Diagram

Diagram aktivitas atau *activity* diagram adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan workflow (aliran kerja) dalam banyak kasus. *Activity* diagram mempunyai peran seperti halnnya *flowcar*t, akan tetapi perbedaannya dengan flowchart adalah activity diagram bisa mendukung perilaku paralel sedangkan flowcart tidak bisa [12].

Dibawah ini adalah contoh beberapa simbol yang umum dipakai dan tentu digunakan dalam penelitian penulis, sebagai berikut [12]:



(Sumber : Pemodelan Visual dengan UML [Munawar])

Gambar 2. 11 Contoh simbol umum *activity* diagram

2.12 *Sequence* Diagram

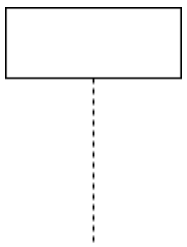
Diagram *sequence* atau *Sequence* Diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan sejumlah contoh objek dan *message* (pesan) yang diletakkan diantara obyek-obyek ini didalam *use case*.

Komponen utama *sequence* diagram terdiri atas obyek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukkan dengan *progress vertical* [12]..

1) Obyek/*Participant*

   Obyek diletakkan di dekat bagian atas diagram dengan urutan dari kiri ke kanan. Mereka diatur dalam urutan guna menyederhanakan diagram. Pengertian obyek hanya ada di UML 1, sedangkan di UML 2 istilah obyek diganti dengan *participant* [12].

   Setiap *participant* terhubung dengan garis titil-titik yang disebut *lifeline*. Sepanjang *lifeline* ada kotak yang disebut *activation*. *Activation* mewakili sebuah eksekusi operasi dari *participant.* Panjang kotak ini berbanding lurus dengan durasi *activation.*



(Sumber : Pemodelan Visual dengan UML [Munawar])

Gambar 2. 12 Obyek/*Participant* pada *sequence diagram*

2) *Message*

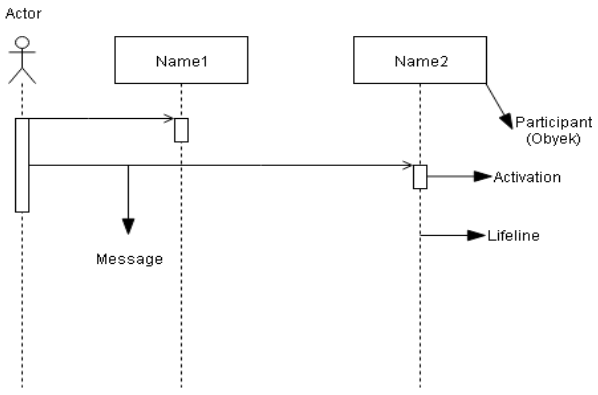
Sebuah *message* bisa jadi *simple*, *synchronous* atau *asynchronous*. *Message* yang simpel adalah sebuah perpindahan (transfer) kontrol dari satu *participant* ke *participant* yang lainnya. Jika sebuah atas *message* tersebut akan ditunggu sebelum diproses dengan urusannya. Namun jika *message asynchronous* yang dikirimkan, maka jawaban atas *message* pada *sequence diagram* bisa dilihat sebagai berikut [12]:

(Sumber : Pemodelan Visual dengan UML [Munawar])

Gambar 2. 13 Simbol-simbol *message*

3) *Time*

Time adalah diagram yang mewakili waktu pada arah *vertical*. Waktu dimulai dari atas kebawah. *Message* yang lebih dekat dari atas akan dijalankan terlebih dahulu dibanding *message* yang lebih dekat kebawah [13].



(Sumber : Pemodelan Visual dengan UML [Munawar])

Gambar 2. 14 Simbol-simbol yang ada pada *sequence diagram*

Berikut Tabel 2.4 Operator umum yang sering digunakan di *interaction frame*:

Tabel 2. 3 Operator-operator umum pada *interaction frame*

Operator	Keterangan
Alt	Alternatif fragment. Hanya kondisi True yang akan dijalankan
Opt	Optional fragment. Jika kondisi mendukungnya True
Par	Paralel. Setiap fragment dijalankan secara paralel
Loop	Looping. Dijalankan berulang kali, guard menunjukan basis iterasi
Region	Fragment hanya punya satu thread untuk menjalankannya
Neg	Negative, fragment menunjukkan interaction yang salah
Ref	Rerences, menunjukkan interaksi pada diagram class lain
Sd	Sequence diagram