

BAB II LANDASAN TEORI

2.1 Penelitian Terkait

Beberapa penulisan yang pernah dilakukan sebelumnya mengenai media pembelajaran adalah:

- a. “PERANCANGAN APLIKASI ANDROID SEBAGAI MEDIA PEMBELAJARAN MATEMATIKA PADA MATERI DIMENSI TIGA UNTUK SISWA SMA KELAS X” (Rohmi Julia Purbasari: 2013), aplikasi yang dibuat berupa aplikasi berbasis *android* yang memuat materi dimensi tiga. Selain dapat dioperasikan pada perangkat android, aplikasi ini juga dapat dioperasikan pada komputer atau laptop yang berbasis *Windows*.
- b. “PERANCANGAN APLIKASI MEDIA PEMBELAJARAN FISIKA PADA POKOK BAHASAN FLUIDA STATIS UNTUK SISWA SMA BERBASIS ANDROID” (M. Iqbal: 2016), penelitian ini termasuk ke dalam penelitian *Research and Develoment* () yang digunakan untuk menghasilkan produk tertentu. Aplikasi yang dibuat berupa aplikasi berbasis *android* yang membahas fisika pada fluida statis untuk SMA. Pembuatan aplikasi tersebut menggunakan perangkat lunak android studio yang dijalankan pada sistem operasi *windows*.
- c. “APLIKASI PENGENALAN TEKNIK DASAR KARATE BERBASIS ANDROID” (Muhaimin), aplikasi yang dibuat berupa aplikasi berbasis *android* yang membahas tentang tutorial teknik karate. Aplikasi ini dirancang dengan memanfaatkan aplikasi *Eclipse* juno sebagai perangkat penunjang utama

pembuatan aplikasi *android* serta *adobe photoshop cs 5* sebagai penunjang pembuatan animasi. Aplikasi ini dapat dipergunakan secara offline.

2.2 Teori Umum

Teori umum adalah teori yang berkaitan dengan dunia teknologi informasi dari komputer.

2.2.1 Perancangan

Perancangan sistem adalah sekumpulan aktivitas yang menggambarkan secara rinci bagaimana sistem akan berjalan. Hal itu bertujuan untuk menghasilkan produk perangkat lunak yang sesuai dengan kebutuhan user (Satzinger dkk, 2010). Perancangan sistem adalah sebuah kegiatan merancang dan menentukan cara mengolah sistem informasi dari hasil analisa sistem sehingga dapat memenuhi kebutuhan dari pengguna termasuk diantaranya perancangan user interface, data, dan aktivitas proses (O'Brian dan Marakas, 2010).

Dari pengertian ahli di atas, penulis menyimpulkan bahwa perancangan adalah aktivitas merumuskan suatu konsep dan ide yang baru untuk suatu sistem atau aplikasi yang akan berjalan nantinya.

2.2.2 Aplikasi

Aplikasi merupakan suatu program yang dibuat oleh pemakai yang ditujukan untuk melakukan suatu tugas khusus (Kadir, 2003: 121).

Menurut kadir dikelompokkan menjadi 2 bagian, yaitu:

a. Program Aplikasi Serbaguna

Program aplikasi serba guna adalah program aplikasi yang dapat digunakan oleh pemakai untuk melaksanakan hal-hal yang bersifat umum. Misalnya untuk membuat dokumentasi untuk mengirimkan surat secara elektronik, dan untuk mengotomatisasikan tugas-tugas individu yang bersifat berulang, seperti untuk melakukan perhitungan-perhitungan yang bersifat rutin. Termasuk dalam kategori ini antara lain adalah DBMS sederhana, surat elektronik, pengolah kata, lembar kerja dan program presentasi, aplikasi serbaguna seringkali disebut perangkat lunak pemakai akhir.

b. Program Aplikasi Spesifik

Aplikasi spesifik merupakan aplikasi yang ditujukan untuk menangani hal-hal yang spesifik. Misalnya program pada sistem POS (*Point Of Sale*) dan ATM, termasuk dalam kategori ini adalah program yang disebut sebagai paket aplikasi atau perangkat lunak paket. Contohnya adalah *Microsoft Office* dan *OpenOffice.org* yang menggabungkan suatu aplikasi pengolah kata, lembar kerja, serta beberapa aplikasi lainnya. Aplikasi-aplikasi dalam satu paket biasanya memiliki antar muka pengguna yang memiliki kesamaan sehingga memudahkan pengguna untuk mempelajari dan menggunakan tiap aplikasi.

2.2.3 Media Pembelajaran

Media pembelajaran yang ada sekarang terdiri dari beberapa bentuk seperti bentuk buku, *e-learning (website)*, dan *mobile learning*. Penggunaan media pembelajaran dalam bentuk *mobile* memiliki beberapa kelebihan

dibandingkan bentuk yang lainnya, di antaranya sebagai sebuah media yaitu suatu media yang menghubungkan unsur edukasi (*education*) dengan hiburan (*entertainment*) atau belajar sambil bermain, mudah untuk dibawa kemana-mana karena terinstal dalam *smartphone*, dan dalam penggunaannya karena tidak menghabiskan terlalu banyak waktu (Lestari, 2013). Pembelajaran diartikan sebagai proses penciptaan lingkungan yang memungkinkan terjadinya proses belajar. Jadi dalam pembelajaran yang utama adalah bagaimana siswa belajar (Sodikin, Noersasongko, & Pramudi, 2009).

Dari pendapat di atas disimpulkan bahwa media pembelajaran adalah segala sesuatu yang dapat menyalurkan pesan, dapat merangsang pikiran, perasaan, dan kemauan peserta didik sehingga mendorong terciptanya proses belajar pada diri peserta didik.

Ada beberapa media pembelajaran, diantaranya:

- a. *Media Visual*: grafik, diagram, chart, bagan, poster kartun, komik.
- b. *Media Audial*: radio, *tape recorder*, labolatorium bahasa, dan sejenisnya.
- c. *Projected Still Media*: *slide*; *over head projector* (OHP), *in focus* dan sejenisnya.
- d. *Projected Motion Media*: film, televisi, video (VCD, DVD, VTR), komputer dan sejenisnya.

Media pembelajran juga bisa kita artikan sebagai sarana panduan untuk suatu proses pembelajaran. Menurut KBBI (Kamus Besar Bahasa Indonesia), yang mempunyai arti penunjuk jalan, pengiring, dan buku

petunjuk; khusus diterbitkan dengan bentuk dan teknik penyajian isi yang praktis.

Menurut KBBI panduan adalah suatu penyajian informasi dan memandu atau memberikan tuntunan kepada pembaca untuk melakukan apa yang disampaikan di dalam buku tersebut (<http://kbbi.web.id/>).

2.2.4 Olahraga

Menurut KBBI olahraga adalah gerak badan untuk menguatkan dan menyehatkan tubuh (seperti sepak bola, berenang, lempar lembing) (<http://kbbi.web.id/>).

Olahraga adalah kegiatan jasmani yang dilakukan untuk menaikkan suhu tubuh, sehingga otot-otot yang kaku akan melemas kembali. Olahraga adalah kegiatan yang seharusnya wajib masuk dalam jadwal aktivitas karena manfaatnya sangat besar untuk tubuh manusia.

2.2.5 Android

Android adalah sebuah system operasi untuk perangkat *mobile* berbasis *linux* yang mencakup sistem operasi, *middleware*, dan aplikasi. Android menyediakan platform bagi para pengembang untuk menciptakan aplikasi mereka. Awalnya google inc, membeli android inc, yang merupakan pendukung baru yang membuat piranti lunak untuk ponsel/handphone. Kemudian untuk mengembangkan android, dibutuhkan *Open Handset Alliance*, piranti perangkat keras dan lunak, dan telekomunikasi termasuk *google*, *HTC*, *Intel*, *Motorola*, *Qual-comm*, *T-mobile*. Dan Nvidia (Martiniyanti & lauren, 2013: 2-4).

a. Sejarah Android

Pada saat pertama perilisan perdana android, 5 November 2007, android bernama *Open Handset Alliance* menyatakan mendukung perkembangan *open source* pada perangkat *mobile*. Di pihak lain, Google merilis kode-kode android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan *open platform* perangkat seluler. Di dunia ini terdapat dua jenis distributor sistem operasi android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mobile Service* (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD).

Sekitar September 2007, sebuah studi melaporkan bahwa Google mengajukan hak paten aplikasi telepon seluler. Selain itu, Google mengenalkan *Nexus One*, salah satu jenis *smartphone* yang menggunakan *Android* sebagai sistem operasinya. Telepon seluler ini diproduksi oleh HTC Corporation dan tersedia dipasaran pada 5 Januari 2010.

Pada masa saat ini kebanyakan vendor-vendor *smartphone* sudah memproduksi *smartphone* berbasis android, vendor-vendor itu antara lain *HTC, Motorola, Nexian, Samsung, LG, HKC, Huawei, Archos, Webstation Camangi, Dell, Nexus, SciPhone, WayteQ, Sony Ericsson, Accer, Philips, T-Mobile, IMO, Asus*, dan masih banyak lagi vendor *smartphone* di dunia yang memproduksi android, hal ini karena android itu adalah sistem operasi yang *open source* sehingga bebas didistribusikan dan dipakai oleh vendor manapun.

Tidak hanya menjadi sistem operasi di *smartphone*, saat ini android menjadi pesaing utama dari *Apple* pada sistem operasi Tablet PC. Pesatnya pertumbuhan android selain *factor* yang disebutkan di atas adalah karena android itu sendiri merupakan *platform* yang sangat lengkap baik itu sistem operasinya, Aplikasi dan *Tool* pengembangan, *market* aplikasi android serta dukungan yang sangat tinggi dari komunitas *open source* di dunia, sehingga android terus berkembang baik dari segi teknologi, maupun dari segi jumlah *device* yang ada di dunia (Safaat, 2011: 1-3)

b. Android SDK (Software Development Kit)

Android SDK (*Software Development Kit*) adalah *tools* API (*Application Programming Interface*) yang diperlukan untuk mengembangkan aplikasi pada *platform* android menggunakan bahasa pemrograman java. Android merupakan *subset* perangkat lunak untuk ponsel meliputi sistem operasi, *middleware*, dan aplikasi kunci yang di *release* oleh Google. Saat ini disediakan Android SDK (*Software Development Kit*) sebagai alat bantu dan API untuk mulai mengembangkan aplikasi, android memberikan anda kesempatan untuk membuat aplikas yang kita butuhkan bukan merupakan aplikasi bawaan *handphone/ smartphone* (Safaat, 2011: 5-6).

Menurut Murtiwiati & Lauren secara garis besar, arsitektur android dapat dijelaskan dan digambarkan sebagai berikut:

- a. *Application* dan *widgets*, adalah *layer* di mana berhubungan dengan aplikasi saja, di mana biasanya *download* aplikasi dijalankan kemudian dilakukan instalasi dan jalankan aplikasi tersebut.

- b. *Application Frameworks*, adalah *layer* di mana para pembuat aplikasi melakukan pengembangan/pembuatan aplikasi yang akan dijalankan di sistem operasi android, karena pada *layer* inilah aplikasi dapat dirancang dan dibuat, seperti *content-providers* yang berupa SMS dan panggilan telepon.
- c. *Libraries*, *layer* di mana fitur-fitur android berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya. Berjalan di atas kernel, *layer* ini mengikuti berbagai *libraries* C/C++ inti seperti Libc dan SSL.
- d. *Android Run Time*, *layer* yang membuat aplikasi android dapat dijalankan di man dalam prosesnya menggunakan Implementasi Linux.
- e. *Linux Kernel*, adalah *layer* di mana inti dari operating sitem dari android itu berada. Berisi file-file sistem yang mengatur sitem *processing*, memori, *resource*, *drivers*, dan sisitem-sistem operasi android lainnya. Linux Kernel yang digunakan android adalah Linux Kernel *release* 2.6..

2.3 Teori Likert

Menurut Sugiyono (2009) skala Likert digunakan untuk mengukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang tentang fenomena sosial. Fenomena sosial disini adalah tanggapan responden mengenai media pembelajaran tata bahasa Jepang yaitu Sakura Bunpou yang penulis buat. Berikut adalah langkah-langkah penilaian untuk menghitung hasil Angket.


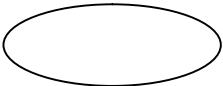
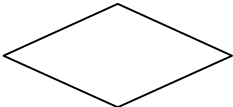
2.4 Teori Perancangan Basis Data

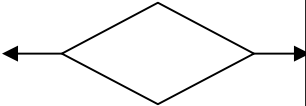
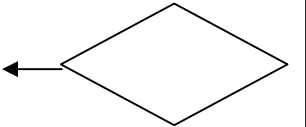
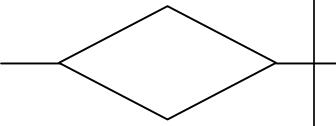
Basis data adalah kumpulan data yang saling berelasi, relasi tersebut biasanya ditunjukkan dengan kunci dari tiap file yang ada. Satu database menunjukan satu kumpulan data yang dipakaidalam satu lingkup perusahaan, instansi (Kristanto & Ir Harianto, 2004).

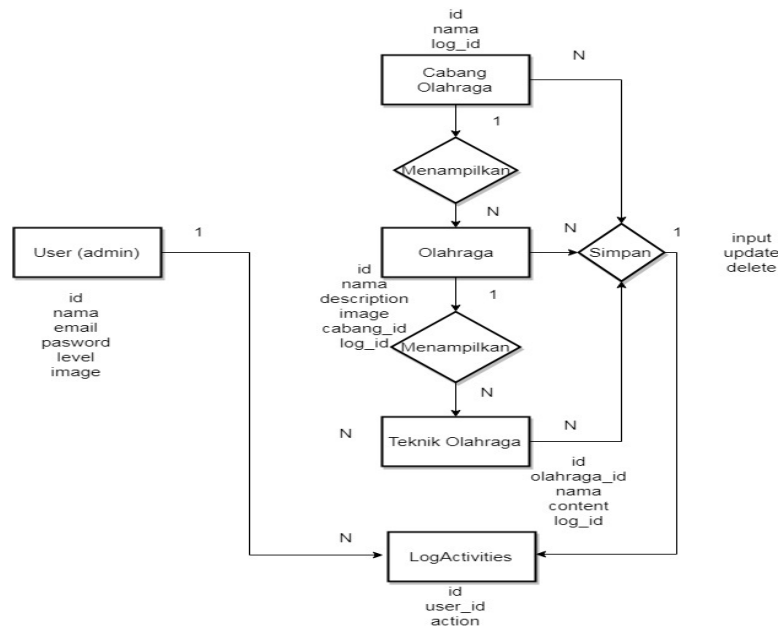
2.4.1 *Entity Relationship Diagram (ERD)*

Entity Relationship Diagram (ERD) merupakan teknik yang digunakan untuk memodelkan kebutuhan data dari suatu organisasi, biasanya dilakukan oleh sistem analis dalam tahap analisis persyaratan (*requirement analysis*) proyek pengembangan sistem. Sementara seolah-olah teknik diagram atau alat peraga memberikan dasar untuk desain relasi *database* yang mendasari sistem informasi yang dikembangkan (Brady dan Loonam, 2010).

Tabel 2.1 Komponen-Komponen ERD

Notasi	Komponen	Keterangan
	Entitas	Individu yang mewakili suatu objek dan dapat dibedakan dengan objek yang lain.
	Atribut	Properti yang dimiliki oleh suatu entitas, dimana dapat mendeskripsikan karakteristik dari entitas tersebut.
	Relasi	Menunjukkan hubungan diantara sejumlah entitas yang berbeda.

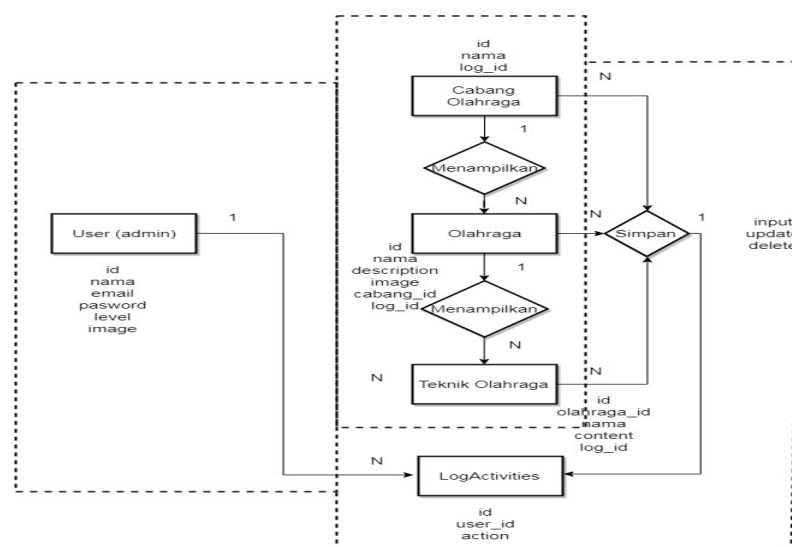
	Relasi 1 : 1	Relasi yang menunjukkan bahwa setiap entitas pada himpunan entitas pertama berhubungan dengan paling banyak satu entitas pada himpunan entitas kedua.
	Relasi 1 : N	Relasi yang menunjukkan bahwa hubungan antara entitas pertama dengan entitas kedua adalah satu banding banyak atau sebaliknya. Setiap entitas dapat berelasi dengan banyak entitas pada himpunan entitas yang lain.
	Relasi N : N	Hubungan ini menunjukkan bahwa setiap entitas pada himpunan entitas yang pertama dapat berhubungan dengan banyak entitas pada himpunan entitas yang kedua, demikian juga sebaliknya.



Gambar 2.1 Komponen-Komponen ERD

2.4.2 Transformasi ERD ke *Logical Record Structure (LRS)*

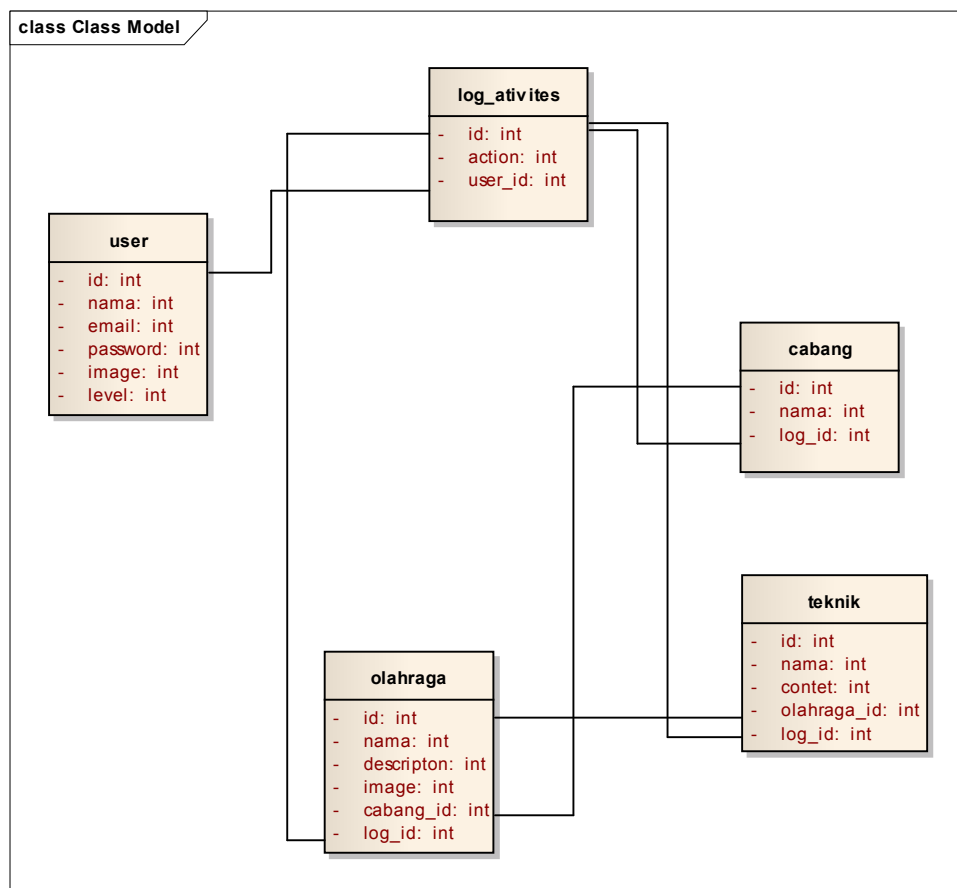
Logical Record Structure (LRS) sebuah model sistem yang digambarkan dengan sebuah diagram ER atau ERD akan mengikuti pola atau aturan permodelan tertentu dalam kaitannya dengan konvensi ke LRS (Hasugian dan Shidiq, 2012).



Gambar 2.2 Contoh Tranformasi ERD ke LRS

2.4.3 Logical Record Structure (LRS)

Logical Record Structure (LRS) adalah representasi dari struktur *record-record* pada hasil relasi antar himpunan entitas. Menentukan kardinalitas, Jumlah Tabel, dan *Foreign Key (FK)*. Setelah ERD ditransformasikan ke bentuk LRD, maka hasil akhir dari proses transformasi tersebut adalah sebuah diagram yang sudah dapat menggambarkan basis data yang akan digunakan. LRS terdiri dari tipe *record*, yang berupa sebuah persegi dengan *field* yang dibutuhkan didalamnya. LRS terdiri juga dari hubungan antara tipe *record* tersebut.



Gambar 2.3 Contoh LRS Diagram

2.4.4 Spesifikasi Basis Data

Basis Data merupakan kumpulan dari data yang saling berhubungan satu sama lain, tersimpan di perangkat komputer dan digunakan oleh perangkat lunak untuk memanipulasinya.

2.5 Definisi *Unified Modelling Language* (UML)

UML disebut sebagai bahasa pemodelan bukan metode. Kebanyakan metode terdiri paling sedikit prinsip, bahasa permodelan dan proses. Bahasa pemodelan (sebagian besar grafik) merupakan notasi dari metode yang digunakan untuk mendesain secara cepat. UML merupakan bahasa standar untuk penulisan *blueprint software* yang digunakan untuk visualisasi, spesifikasi, pembentukan, dan pendokumentasian alat-alat dari sistem perangkat lunak. UML menyediakan notasi-notasi yang membantu memodelkan sistem dari berbagai perspektif. UML tidak hanya digunakan dalam pemodelan perangkat lunak, namun hampir dalam semua bidang yang membutuhkan pemodelan (Sari, 2004: 0-4).

Menurut Sari, UML terbagi menjadi beberapa bagian-bagian utama, yaitu *view*, *diagram*, *model element*, dan *general mechanism*. Di sini akan kita bahas mengenai bagian-bagian diagram dari UML. Diagram berbentuk grafik yang menunjukkan symbol element model yang disusun untuk mengilustrasikan bagian atau aspek tertentu dari sistem. Sebuah diagram merupakan bagian dari suatu *view* tertentu dan ketika digambarkan biasanya dialokasikan untuk *view* tertentu.

2.5.1 *Use Case Diagram*

Use case Diagram adalah diagram yang menggambarkan kebutuhan sistem dari sudut pandang *user* dan merupakan pola perilaku sistem.

Digunakan untuk menggambarkan hubungan antara internal sistem dan eksternal sistem, atau hubungan antara sistem dan *actor*.

Simbol-simbol yang digunakan pada *Use case* Diagram yaitu :

a. *Actor*

Actor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasi *actor*, 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 *actor* berinteraksi dengan *use case*, tetapi tidak memiliki kontrol terhadap *use case*.

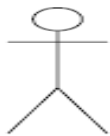
b. *Use Case*

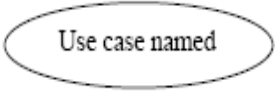

Use case menggambarkan perilaku, termasuk didalamnya interaksi antara *actor* dengan sistem. *Use case* dibuat berdasarkan keperluan *actor*, merupakan “Apa” yang dikerjakan sistem bukan “Bagaimana” sistem mengerjakannya.

c. *Relationship*

Relationship menggambarkan relasi-relasi yang terjadi pada *use case* diagram bisa antara *actor* dengan *use case* atau antara *use case*.

Tabel 2.2 Komponen-Komponen *Use Case*

Simbol	Nama	Keterangan
	<i>Actor</i>	<i>Actor</i> adalah pengguna sistem. <i>Actor</i> tidak terbatas hanya manusia saja, jika sebuah sistem berkomunikasi dengan aplikasi

		lain dan membutuhkan input atau memberikan <i>output</i> , maka aplikasi tersebut juga bisa dianggap sebagai <i>actor</i> .
	<i>Use Case</i>	<i>Use case</i> digambarkan sebagai lingkaran <i>elips</i> dengan nama <i>use case</i> dituliskan didalam <i>elips</i> tersebut.
	<i>Relationship</i>	<i>Asosiasi</i> digunakan untuk menghubungkan <i>actor</i> dengan <i>use case</i> . <i>Asosiasi</i> digambarkan dengan sebuah garis yang menghubungkan antara <i>Actor</i> dengan <i>Use Case</i> .







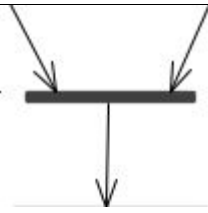


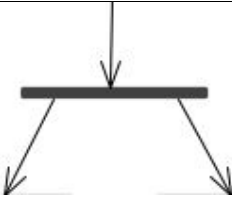
Gambar 2.4 Contoh *Use Case* Diagram

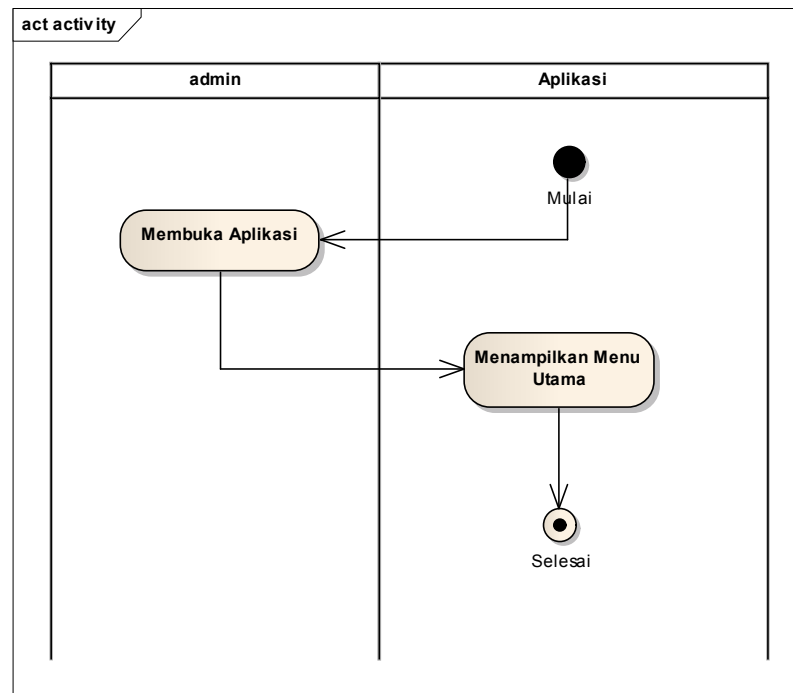
2.5.2 *Activity* Diagram

Activity Diagram menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alur berawal, *decision* yang mungkin terjadi dan bagaimana mereka berakhir. *Activity Diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity Diagram* merupakan *state diagram* khusus, dimana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*).

Tabel 2.3 Komponen-komponen *Activity Diagram*

No.	Simbol	Keterangan
1.		<i>Start</i> merupakan status awal sistem, sebuah diagram aktivitas memiliki sebuah status awal.
2.		<i>Activity</i> merupakan aktivitas yang dilakukan sistem, biasanya diawali dengan kata kerja.
3.		<i>Decision</i> merupakan asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4.		<i>Swimlane</i> digunakan untuk memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.
5.		<i>Finish</i> merupakan status akhir sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
6.		Asosiasi penggabungan dimana lebih dari satu aktivitas gabungan menjadi satu.
7.		<i>Join</i> digunakan untuk menunjukkan kegiatan yang digabungkan.

8.		<p><i>Fork</i> digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel.</p>
----	---	--




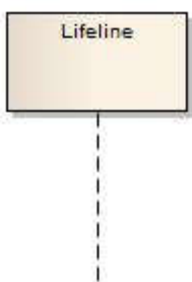

Gambar 2.5 Contoh *Activity* Diagram

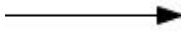
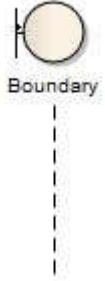


2.5.3 *Sequence* Diagram

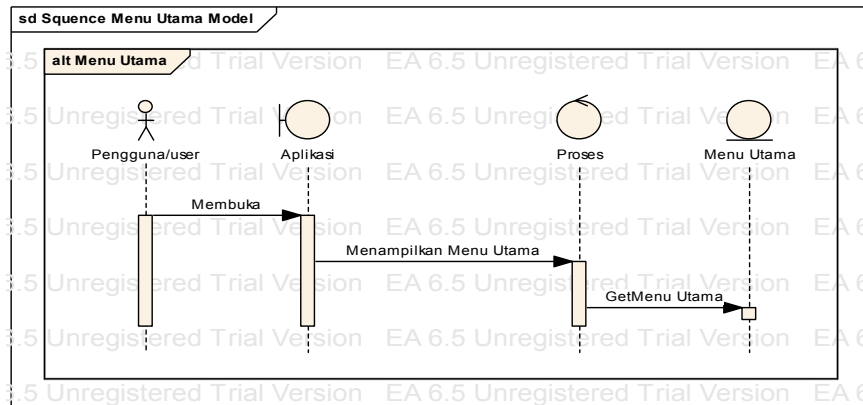
Sequence diagram menggambarkan interaksi antar obyek didalam dan disekitar sistem (termasuk penggunaan, *display* dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (obyek-obyek yang terkait). *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respon dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang memicu aktivitas

tersebut, proses dan perubahan apa saja yang terjadi secara internal dan keluaran apa yang dihasilkan.

Tabel 2.4 Keterangan simbol-simbol *Sequence Diagram*

No.	Simbol	Keterangan
1.		<i>Actor</i> adalah pesan dari seseorang atau sistem lain yang bertukar informasi dengan sistem yang lainnya, kemudian <i>lifeline</i> berhenti atau mulai pada titik yang tepat.
2.		<i>Object Life Line</i> menunjukkan keberadaan dari sebuah objek terhadap waktu. Yaitu objek dibuat atau dihilangkan selama suatu periode waktu diagram ditampilkan, kemudian <i>lifeline</i> berhenti atau mulai pada titik yang tepat.
3.		<i>Activation</i> menampilkan periode waktu selama objek atau aktor melakukan aksi. Dalam <i>object life line</i> , <i>activation</i> berada diatas <i>lifeline</i> dalam bentuk kotak persegi panjang, bagian atas dari kotak merupakan inisialisasi waktu dimulainya suatu kegiatan dan yang dibawah merupakan akhir dari waktu.

4.		<p><i>Message</i> adalah komunikasi antar objek yang membawa informasi dan hasil pada sebuah aksi. <i>Message</i> menyampaikan dari <i>lifeline</i> sebuah objek kepada <i>lifeline</i> yang lainnya, kecuali pada kasus sebuah <i>message</i> dari oboek kepada objek itu sendiri atau dengan kata lain <i>message</i> dimulai dan diakhiri pada <i>lifeline</i> yang sama.</p>
5.		<p><i>Boundary</i> adalah yang digunakan untuk menggambarkan sebuah form.</p>
6.		<p><i>Control</i> digunakan untuk menghubungkan <i>boundary</i> dengan tabel.</p>
7.		<p><i>Entity</i> diguanakan untuk menggambarkan hubungan kegiatan yang akan dilakukan.</p>

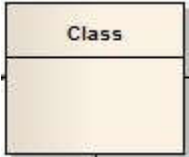

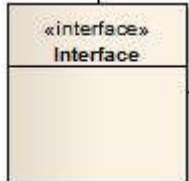






Gambar 2.6 Contoh *Sequence Diagram*

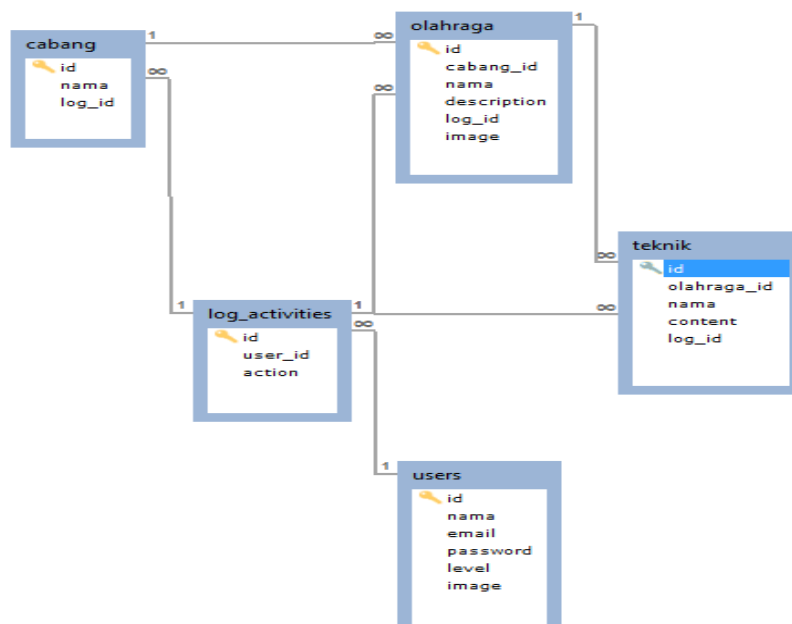
2.5.4 *Class Diagram*

Class Diagram menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi dan lain-lain.

Tabel 2.5 Komponen-komponen *Class Diagram*

No.	Simbol	Keterangan
1.		<i>Class</i> digambarkan dengan bentuk persegi panjang yang dibagi kedalam ruang-ruang terpisah yang terdiri dari mana <i>class</i> , atribut dan operasi-operasinya.
2.		<i>Association</i> adalah representasi/gambaran relasi statis di antara <i>class-class</i> .
3.		<i>Interface</i> sama dengan <i>interface</i> dalam pemrograman berorientasi objek.

4.		<i>Association</i> berarah merupakan relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i> .
5.		<i>Generalisasi</i> merupakan relasi antar kelas dengan makna <i>generalisasi-spesialisasi</i> (umum khusus)
6.		<i>Depedency</i> merupakan relasi antar kelas dengan makna ketergantungan antar kelas.
7.		<i>Aggregation</i> merupakan relasi antar kelas dengan makna semua bagian (<i>whole-part</i>).

Gambar 2.7 Contoh *Class Diagram*

2.6 Aplikasi Pendukung

Dalam perancangan aplikasi ini terdapat beberapa aplikasi yang mendukung untuk pembuatannya agar bisa berjalan dengan baik.

2.6.1 Ionic Framework

Ionic adalah *framework* yang dikhususkan untuk membangun aplikasi *mobile hybrid* dengan HTML5, CSS dan *AngularJS*. *Ionic* menggunakan Node.js, SASS, *AngularJS* sebagai *engine*-nya. *Ionic* dilengkapi dengan komponen-komponen CSS seperti *button*, *list*, *card*, *form*, *grids*, *tabs*, dan masih banyak lagi. Jadi *Ionic* itu merupakan teknologi yang bisa digunakan untuk membuat suatu aplikasi *mobile*. Karena *hybrid* maka aplikasi hanya dibuat 1 kali tetapi sudah *web* bisa dirilis di lebih dari 1 platform alias *cross-platform*. *Ionic* Menggunakan lisensi *Opensource*, menggunakan teknologi *web* terbaru *Ionic* memanfaatkan *AngularJS* untuk implementasi *logica*nya yang menawarkan performa dan respon cepat serasa aplikasi *native*. Selain *Ionic Framework* yang *free* dan *opensource*, *Ionic* juga memperkenalkan *ionic Platform*. *Ionic Platform* memberikan layanan tambahan bagi para *developer ionic*. *Ionic Platform* menambahkan fitur *Create*, *deploy*, *update*, *Analytics*, *Marketing (Ionic Market)* dan *Push notification* kedalam aplikasi yang dibangun dengan *ionic*.

Untuk menunjang *porting* dari aplikasi *web* ke *mobile ionic* menggunakan *cordova(apache cordova)* yaitu sebuah *framework* untuk membangun aplikasi *android* dengan menggunakan *html*, *css* dan *javascript*. *Cordova* dulunya bernama *phonegap* yang dibuat oleh perusahaan bernama *Nitobi*. Pada tahun 2011 *adobe system* membeli *nitobi* dan sekarang

pengembangannya diserahkan kepada *Apache software foundation* sehingga namanya berubah dari *phonegap* menjadi *apache cordova*.

2.6.2 SQLyog

SQLyog adalah salah satu tool administrasi untuk *database MySQL*. Jika kita biasanya menggunakan *PhpMyAdmin* yang include di dalam aplikasi *Xampp* untuk melakukan administrasi *database*, *SQLyog* adalah aplikasi alternatif untuk melakukan proses administrasi *database MySQL*. Banyak fitur yang disediakan oleh *SQLyog* yang tidak disediakan oleh *PhpMyAdmin* maupun *tool* administrasi *database* lainnya seperti *MySQLQueryBrowser*. Dengan *SQLyog* kita dapat membuat *Store Prosedure*, *Function* maupun *Trigger* dengan mudah.

2.6.3 JavaScript

JavaScript adalah bahasa pemrograman web yang bersifat *Client Side Programming Language*. *Client Side Programming Language* adalah tipe bahasa pemrograman yang pemrosesannya dilakukan oleh *client*. Aplikasi *client* yang dimaksud merujuk kepada *web browser* seperti *Google Chrome* dan *Mozilla Firefox*.

Bahasa pemrograman *Client Side* berbeda dengan bahasa pemrograman *Server Side* seperti *PHP*, dimana untuk *server side* seluruh kode program dijalankan di sisi server. Untuk menjalankan **JavaScript**, kita hanya membutuhkan aplikasi *text editor* dan *webbrowser*. *JavaScript* memiliki fitur: *high-level programming language*, *client-side*, *loosely typed* dan berorientasi objek.

JavaScript pada awal perkembangannya berfungsi untuk membuat interaksi antara user dengan situs web menjadi lebih cepat tanpa harus menunggu pemrosesan di *web server*. Sebelum *javascript*, setiap interaksi dari user harus diproses oleh *web server*.

Bayangkan ketika kita mengisi *form registrasi* untuk pendaftaran sebuah situs *web*, lalu men-klik tombol *submit*, menunggu sekitar 20 detik untuk *website* memproses isian *form* tersebut, dan mendapati halaman yang menyatakan bahwa terdapat kolom *form* yang masih belum diisi.

Untuk keperluan seperti inilah *JavaScript* dikembangkan. Pemrosesan untuk mengecek apakah seluruh *form* telah terisi atau tidak, bisa dipindahkan dari *web server* ke dalam *web browser*.

Dalam perkembangan selanjutnya, *JavaScript* tidak hanya berguna untuk *validasi form*, namun untuk berbagai keperluan yang lebih modern. Berbagai animasi untuk mempercantik halaman web, fitur chatting, efek-efek modern, games, semuanya bisa dibuat menggunakan *JavaScript*.

2.6.4 PHP

PHP (*Hypertext Preprocessor*) adalah bahasa pemrograman yang berfungsi untuk membuat website dinamis maupun aplikasi web. Berbeda dengan HTML yang hanya bisa menampilkan konten statis, PHP bisa berinteraksi dengan *database*, *file* dan folder, sehingga membuat PHP bisa menampilkan konten yang dinamis dari sebuah *website*. *blog*, toko *online*, *cms*, forum, dan *websitesocial networking* adalah contoh aplikasi web yang bisa dibuat oleh PHP. Dalam halaman HTML, dapat dimasukkan kode PHP yang akan dieksekusi setiap kali halaman web tersebut diakses. Kode PHP ini

akan diterjemahkan oleh *web server* dan akan dijalankan bersama dengan HTML atau *output* lainnya, yang akan dilihat oleh *user* situs *web*.

2.6.5 MySQL

MySQL adalah sebuah system manajemen *database* relasi (*relational database management system*) yang bersifat terbuka (*open source*). Terbuka maksudnya adalah MySQL boleh di download oleh siapa saja, baik versi kode program aslinya (*source code program*) maupun versi binernya (*executable program*) dan bisa digunakan secara gratis baik untuk dimodifikasi sesuai dengan kebutuhan seseorang maupun sebagai suatu program aplikasi komputer.

2.6.6 Xampp

XAMPP dikembangkan dari sebuah tim proyek bernama *Apache Friends*, yang terdiri dari Tim Inti (*Core Team*), Tim Pengembang (*Development Team*) dan Tim Dukungan (*Support Team*).

XAMPP adalah singkatan yang masing-masing hurufnya adalah :

a. X

X = Program ini dapat dijalankan dibanyak sistem operasi, seperti *Windows*, *Linux*, *Mac OS*, dan *Solaris*.

b. A

A = *Apache*, adalah aplikasi *web server*. Tugas utama *Apache* adalah menghasilkan halaman *web* yang benar kepada *user* berdasarkan kode PHP yang dituliskan oleh pembuat halaman *web*. Jika diperlukan juga berdasarkan kode PHP yang dituliskan, maka dapat saja suatu *database* diakses terlebih dahulu (misalnya dalam *MySQL*) untuk mendukung halaman *web* yang

dihasilkan.

c. M

M = MySQL adalah aplikasi *database* server. Perkembangannya disebut SQL yang merupakan kepanjangan dari *Structured Query Language*. SQL merupakan bahasa terstruktur yang digunakan untuk mengolah *database*. MySQL dapat digunakan untuk membuat dan mengelola *database* beserta isinya.

Kita dapat memanfaatkan MySQL untuk menambahkan, mengubah dan menghapus data yang berbeda dalam *database*.

d. P

P = PHP, bahasa pemrograman *web*. Bahasa pemrograman PHP

2.7 Pengujian Sistem

Pengujian sistem dilakukan dengan mencoba semua kemungkinan yang segala terjadi dan pengujian menggunakan pengujian *black box*. Jika dalam pengujian ditemukan kesalahan, maka akan dilakukan penelusuran dan perbaikan untuk memperbaiki kesalahan yang terjadi.

2.7.1 Tujuan Pengujian

Tujuan dalam pengujian dari sebuah perangkat lunak adalah sebagai berikut :

- a. Proses menjalankan program dengan maksud mencari kesalahan (*error*).
- b. Kasus uji yang baik adalah kasus yang memiliki peluang untuk mendapatkan kesalahan yang belum diketahui.

- c. Pengujian dikatakan berhasil bila dapat memunculkan kesalahan yang belum diketahui.
- d. Jadi pengujian yang baik bukan untuk memastikan tidak ada kesalahan tetapi untuk mencari sebanyak mungkin kesalahan yang ada pada program.

2.7.2 Prinsip Pengujian

- a. Semua pengujian harus dapat diurutkan sampai kepada spesifikasi kebutuhan perangkat lunak.
- b. Pengujian harus dimulai dari lingkup yang kecil ke lingkup yang besar.
- c. Pengujian yang mendalam tidak mungkin dilakukan karena tidak mungkin mengeksekusi semua jalur permutasi.
- d. Supaya efektif (memiliki probabilitas yang tinggi dalam menemukan kesalahan), pengujian harus dilakukan oleh pihak lain yang independen.
- e. Pengujian harus direncanakan jauh sebelum dilakukan.

2.7.3 Pengujian *Black Box*

Black box testing adalah pengujian yang dilakukan hanya mengamati hasil eksekusi melalui data uji dan memeriksa fungsional dari perangkat lunak. Jadi dianalogikan seperti melihat suatu kotak hitam. Sama seperti pengujian ini hanya mengevaluasi dari tampilan luarnya (*user interface* nya), pada tahap ini merupakan kelanjutan dari tahap implementasi yaitu melakukan pengujian terhadap aplikasi yang dibangun. Pengujian yang akan dilakukan yaitu dengan pengujian *black box* yang berfokus pada persyaratan fungsional perangkat lunak.