BAB 2
TINJAUAN PUSTAKA

2.1 Profil Tempat Penelitian

Aki No Sora adalah lembaga kursus yang bergerak dalam bidang dan bahasa Jepang. Aki No Sora saat ini menawarkan jasa les atau kursus bahasa Jepang secara privat dan jasa interpretasi untuk menjembatani perusahaan yang memiliki kebutuhan komunikasi dengan klien yang berbahasa Jepang – Inggris – Indonesia.

2.1.1 Sejarah Aki No Sora

Didirikan oleh Aki Pratomodano selaku Founder dan President Director, Aki no Sora berdiri pada tahun 2014 yang bertempat di Jalan Jl. Teuku Angkasa No. 26, Bandung, Jawa Barat. Bermodalkan kemampuan berbahasa Jepang setingkat N1 atau setara dengan *Native speaker* orang Jepang, beliau membuka les bahasa Jepang secara private yang setiap kelas berisi oleh 10 atau lebih orang.

Aki No Sora adalah lembaga kursus yang mengedepankan IT sebagai salah satu fitur andalan mereka. Staff-staff yang ada di lembaga ini memiliki kemampuan IT yang cukup baik, beberapa diantaranya merupakan lulusan dari ITB, UNIKOM. Mengetahui dengan pentingnya teknologi dalam metode pengajaran yang efisien, lembaga ini menyediakan fitur-fitur berbasis IT seperti Online Class dan Virtual Teacher. Keduanya adalah bentuk pengajaran yang dapat dilakukan tanpa harus bertatap muka atau hadir secara fisik atau tidak langsung.

Saat ini, Aki No Sora merupakan lembaga kursus bahasa jepang yang memberikan jasa baik perorangan maupun perusahaan. Tidak hanya les privat yang ditawarkan, tetapi juga jasa interpretasi untuk perusahaan yang memiliki kebutuhan untuk berkomunikasi dengan klien mereka yang berbahasa Jepang

– Inggris – Indonesia. Lembaga kursus ini memiliki website beralamatkan www.akinosora.co

2.1.2 Nama Lembaga

Aki no Sora atau Autumn Sky terdiri dari dua kata, pertama kata *Aki* atau *Autumn* yang diambil dari founder dan President Director lembaga ini, Aki Pratomodano. Sedangkan kata *Sora* atau *Sky* sebagaimana filosofi yang sering dikaitkan dengan ilmu, bahwa langit yang luas dan tidak ada habisnya begitupula dengan ilmu.

2.1.3 Logo Lembaga



Gambar 2.1 Logo Aki No Sora

Spesifikasi dan Makna Logo:

1. Lingkaran merah : Bendera Jepang
2. 秋の空 : terdiri dari 秋 atau Aki (dalam huruf kanji), の atau No, dan 空 atau Sora (dalam huruf kanji).

2.1.4 Visi

Visi merupakan merupakan pernyataan kondisi masa depan yang ingin diraih oleh organisasi, serta menunjukkan bayangan, keinginan, atau cita-cita yang ingin dicapai organisasi di masa depan. Aki no Sora memiliki visi untuk menjadi jembatan antara Jepang, Indonesia dan dunia. perusahaan berusaha untuk meningkatkan kualitas pelayanan dan pendidikan yang diberikan oleh perusahaan dari Kota Bandung di Jawa Barat, Indonesia.

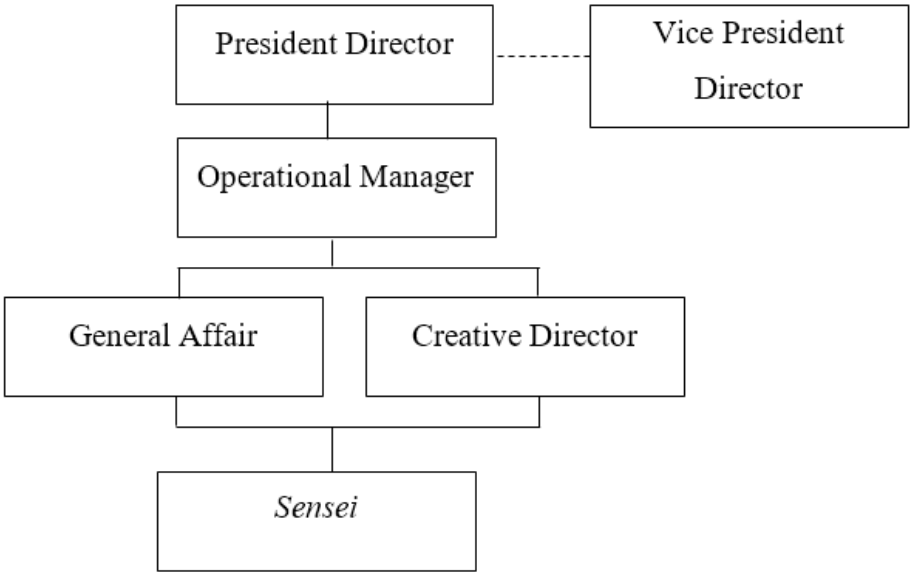
2.1.5 Misi

Misi merupakan pernyataan yang menunjukkan maksud didirikan / dibentuknya organisasi dan lingkup bisnis/kegiatan yang harus dijalankan atau yang justru tidak boleh dijalankan oleh organisasi. Aki No Sora memiliki misi berikut:

1. Menyajikan pendidikan untuk memenuhi kebutuhan berbahasa Jepang maupun pendidikan karakter kepada peserta didik.
2. Menyajikan pendidikan dengan kualitas yang baik melalui pendekatan yang efektif.
3. Memberikan pelayanan berupa interpreter yang sesuai dan tepat dengan kebutuhan klien.

2.1.6 Struktur Organisasi

Dalam sebuah organisasi agar tarcapai susunan kerja yang baik ditiap kinerja anggotanya memerlukan sebuah struktur yang terencana dan dapat memperlihatkan alur kerja yang baik. Adapun struktur organisasi Aki no Sora adalah sebagai berikut :



Gambar 2.2 Struktur Organisasi

2.2 Landasan Teori

Teori pendukung atau landasan teori dalam penyusunan skripsi ini sangat diperlukan karena sebagai referensi untuk menunjang atau memperdalam pemahaman terhadap informasi-informasi yang disajikan.

2.2.1 Bahasa Jepang

Sebelum abad ke-4 Masehi, orang Jepang tidak memiliki sistem menulis mereka sendiri. Selama abad ke-5, mereka mulai mengambil dan mengadopsi tulisan Cina, bersama dengan aspek lain dari budaya Cina, mungkin melalui Korea. Namun, orang-orang Jepang sudah mengetahui tulisan Cina sejak abad 1 Masehi dari karakter yang muncul pada barang impor Cina.

Pada awalnya, orang Jepang menulis dengan Cina klasik atau dalam gaya campuran Jepang-Cina. Contoh dari gaya campuran adalah *kojiki* yang merupakan tulisan antik pada tahun 712 Masehi. Mereka lalu mulai menggunakan karakter Cina untuk menulis tulisan Jepang dengan gaya *man'yogana*, yang menggunakan karakter tersebut untuk arti fonetik.

Seiring dengan waktu, sistem penulisan dengan karakter Cina biasa digunakan untuk kata yang diambil dari Cina atau kosakata Jepang yang memiliki arti kata yang sama atau mirip. Karakter cina juga digunakan untuk arti fonetik untuk menulis elemen bahasa dan karakter-karakter tersebut disederhanakan dan menjadi dua tulisan suku kata, *hiragana* dan *katakana*.

Tulisan Jepang modern ditulis dengan gabungan *hiragana* dan *katakana*, ditambah *kanji*. Tulisan Jepang modern juga memasukkan *romaji*, (tulisan Roman), yang merupakan cara standar dalam menulis tulisan Jepang dengan huruf latin, atau kata-kata non-Jepang.

Berikut merupakan ciri penting dari tulisan Jepang

1. Sistem penulisan : campuran (suku kata *kana* + *semanto-phonetic kanji*)
2. Arah tulisan : dari kanan ke kiri untuk tulisan vertikal atau dari kiri ke kanan untuk tulisan horizontal.

2.2.1.1 Suku Kata *Hiragana*

Hiragana dibentuk dari karakter Cina yang awal mulanya disebut *onnade* atau “tangan perempuan” karena umumnya digunakan oleh perempuan. Sedangkan laki-laki menulis dengan *kanji* dan *katakana*. Pada abad ke-10, *hiragana* digunakan oleh setiap orang.

Pada versi awal *hiragana*, ada banyak karakter yang berbeda untuk menggambarkan suku kata yang sama, namun sistemnya disederhanakan sehingga hanya ada satu ke satu relasi antara suku kata yang diucapkan dan yang ditulis.

Berikut merupakan tabel relasi suku kata yang diucapkan dengan yang tertulis dalam *hiragana* :

Tabel 2.1 Tabel suku kata hiragana

Suku kata dasar					Suku kata tambahan					
あ a	あ i	う u	え e	お o	が ga	ぎ gi	ぐ gu	げ ge	ご go	平仮名
か ka	き ki	く ku	け ke	こ ko	ざ za	じ ji	ず zu	ぜ ze	ぞ zo	
さ sa	し shi	す su	せ se	そ so	だ da	ぢ ji	づ zu	で de	ど do	
た ta	ち chi	つ tsu	て te	と to	ば ba	び bi	ぶ bu	べ be	ぼ bo	
な na	に ni	ぬ nu	ね ne	の no	ぱ pa	ぴ pi	ぷ pu	ぺ pe	ぽ po	
は ha	ひ hi	ふ fu	へ he	ほ ho	きゃ kya	きゅ kyu	きょ kyo	ぎゃ gya	ぎゅ gyu	ぎょ gyo
ま ma	み mi	む mu	め me	も mo	にゃ nya	にゅ nyu	にょ nyo	ひゃ hya	ひゅ hyu	ひょ hyo
や		ゆ		よ	びゃ	びゅ	びょ	ぴゃ	ぴゅ	ぴょ

Suku kata dasar

ya		yu		yo
ら ra	り ri	る ru	れ re	ろ ro
わ wa				を wo
ん n	ひらがな Hiragana			

Suku kata tambahan

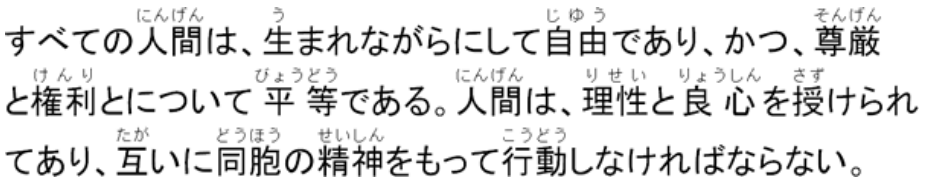
bya	byu	byo	pya	pyu	pyo
みゃ mya	みゅ myu	みょ myo	りゃ rya	りゅ ryu	りょ ryo
じゃ ja	じゅ ju	じぇ je	じょ jo	ちゃ cha	ちゅ chu
ちぇ che	ちょ cho	しゃ sha	しゅ shu	しぇ she	しょ sho

2.2.1.1.1 Karakteristik dan Penggunaan Suku Kata Hiragana

Suku kata hiragana terdiri dari 48 suku kata yang umumnya digunakan untuk menulis akhir kata, atau *okutigana* dalam bahasa Jepang. Hiraga juga banyak digunakan pada bahan atau materi untuk anak-anak, buku, animasi, dan buku komik, untuk menulis kata-kata yang tidak secara umum ditulis dengan *kanji*, seperti kata *adverb* (kata keterangan atau tambahan) dan beberapa kata benda dan *adjectiva*, atau untuk kata-kata dimana *kanji*nya telah tertinggal jaman atau maknanya sudah berubah.

Hiragana juga seringkali ditulis diatas atau bersamaan dengan *kanji* untuk menjelaskan pelafalan. *Hiragana* digunakan dengan cara disebut sebagai *furigana*. Pada teks horizontal, *furigana* yang muncul diatas *kanji* dan pada teks vertikal, *furigana* muncul disebelah kanan *kanji*.

Berikut merupakan contoh dari *furigana*:



Dalam *romaji* atau latin :

Subete no ningen wa, umare nagara ni shite jiyū de ari, katsu, songen to kenri to ni tsuite byōdō de aru. Ningen wa, risei to ryōshin to o sazukerareteari, tagai ni dōhō no seishin o motte kōdōshinakerebanaranai.

Arti :

Semua manusia dilahirkan bebas dan sama dalam martabat dan hak-hak. Mereka dikaruniai akal dan hati nurani dan harus bertindak terhadap satu sama lain dalam semangat persaudaraan.

2.2.1.2 Suku Kata *Katakana*

Suku kata *katakana* berasal dari karakter Cina yang disingkat yang digunakan oleh biarawan Buddha untuk menandakan pelafalan yang benar dari tulisan Cina pada abad ke-9. Nampaknya, urutan dari suku kata berdasarkan pada tulisan *Siddham*. Pada awalnya, ada banyak simbol untuk menggambarkan satu suku kata bahasa Jepang yang diucapkan, namun selama bertahun-tahun sistem

ini kemudian diefisienkan. Pada abad ke-14, ada banyak atau kurang dari satu ke satu relasi antara suku kata yang diucapkan dengan yang tertulis. *Katakana* pada faktanya merupakan bagian yang menggambarkan bagian dari *kanji*.

Berikut merupakan tabel relasi suku kata yang diucapkan dengan yang tertulis dalam *katakana* :

Tabel 2.2 Suku Kata *Katakana*

Suku kata dasar

ア a	イ i	ウ u	エ e	オ o
カ ka	キ ki	ク ku	ケ ke	コ ko
サ sa	シ shi	ス su	セ se	ソ so
タ ta	チ chi	ツ tsu	テ te	ト to
ナ na	ニ ni	ヌ nu	ネ ne	ノ no
ハ ha	ヒ hi	フ fu	ヘ he	ホ ho
マ ma	ミ mi	ム mu	メ me	モ mo
ヤ ya		ユ yu		ヨ yo
ラ ra	リ ri	ル ru	レ re	ロ ro
ワ wa				ヲ wo

Suku kata tambahan

ガ ga	ギ gi	グ gu	ゲ ge	ゴ go	片仮名
ザ za	ジ ji	ズ zu	ゼ ze	ゾ zo	
ダ da	ヂ ji	ヅ zu	デ de	ド do	
バ ba	ビ bi	ブ bu	ベ be	ボ bo	
パ pa	ピ pi	プ pu	ペ pe	ポ po	
キャ kya	キュ kyu	キョ kyo	ギャ gya	ギュ gyu	ギョ gyo
ニャ nya	ニュ nyu	ニョ nyo	ヒャ hya	ヒュ hyu	ヒョ hyo
ビャ bya	ビュ byu	ビョ byo	ピャ pya	ピュ pyu	ピョ pyo
ミャ mya	ミュ myu	ミョ myo	リャ rya	リュ ryu	リョ ryo
ジャ ja	ジュ ju	ジェ je	ジョ jo	チャ cha	チュ chu

Suku kata dasar

ン	カタカナ
n	Katakana

Suku kata tambahan

チェ	チョ	シャ	シュ	シェ	ショ
che	cho	sha	shu	she	sho

ウィ	ウェ	ウォ	チィ	ヂィ	トゥ	ドゥ	ドュ	フィ	フェ	フォ	フュ
wi	we	wo	ti	di	tu	du	dyu	fi	fe	fo	fyu

2.2.1.2.1 Karakteristik dan Penggunaan Suku Kata *Katakana*

Suku kata *katakana* terdiri dari 48 suku kata dan awalnya disebut "tulisan laki-laki". Secak abad ke-20, *katakana* telah digunakan untuk menulis kata serapan non-Cina, kata untuk menggambarkan suara semisal "*bang!*" "*boom!*", nama asing. Sebelum abad ke-20, kata serapan ditulis dengan huruf *kanji*.

Berikut merupakan contoh dari *katakana*:

スベテノニンゲンハ、ウマレナガラニシテジユーデアリ、カツ、ソンゲントケンリトニツイテビョードーデアル。ニンゲンハ、リセイトリョーシントヲサズケラレテオリ、アガイニドーホーノセイシンヲモッテコードーシナケレバナラナイ。

Dalam *romaji* atau latin :

Subete no ningen wa, umare nagara ni shite jiyū de ari, katsu, songen to kenri to ni tsuite byōdō de aru. Ningen wa, risei to ryōshin to o sazukerareteari, tagai ni dōhō no seishin o motte kōdōshinakerebanaranai.

Arti :

Semua manusia dilahirkan bebas dan sama dalam martabat dan hak-hak. Mereka dikaruniai akal dan hati nurani dan harus bertindak terhadap satu sama lain dalam semangat persaudaraan.

2.2.1.3 Karakter atau Huruf Kanji

Antara 5000 dan 10000 karakter atau *kanji*, digunakan untuk menulis bahasa Jepang. Pada tahun 1981, dalam usaha untuk memudahkan dalam membaca dan menulis bahasa Jepang, pemerintah Jepang mengenalkan 常用漢字表 (jōyō kanji hyō) atau "Daftar karakter Cina untuk Penggunaan Umum", yang berisi 1945 karakter biasa atau umum, ditambah karakter untuk nama orang (人名用漢字 - jinmeiyô-kanji). Pada tahun 2010, tambahan 196 *kanji* yang umum digunakan kemudian ditambahan ke dalam list ini sehingga jumlahnya adalah 2136 *kanji*.

Koran, media lain, dan publikasi sebagian besar menggunakan *jōyō kanji* dan memasukkan *furigananya.* Anak-anak Jepang diharapkan untuk mengetahui semua *jōyō kanji* saat menyelesaikan SMA, namun untuk membaca publikasi atau literatur, mereka perlu mengetahui dua atau tiga ribu *kanji.*

Kata *kanji* adalah versi Jepangnya kata Cina *hànzì*, yang memiliki arti "karakter *Han*". *Han* merujuk kepada Dinasti *Han* (206 Sebelum Masehi – 220 Masehi) dan merupakan sebutan untuk orang Cina itu sendiri).

Ketika orang Jepang mengadopsi karakter Cina untuk menulis bahasa Cina, mereka juga mengambil banyak kata-kata Cina yang disebut Sino-Jepang. Sampai saat ini sekitar setengah dari kosa kata Jepang berasal dari Cina dan kanji Jepang digunakan untuk merepresentasikan baik kata dari Cina dan kata asli Jepang yang memiliki arti sama.

Sebagai contoh, kata asli Jepang untuk air adalah mizu dan kata Sino-Jepang adalah *sui.* Keduanya ditulis dengan karakter atau huruf yang sama. Yang pertama tadi, disebut sebagai *kun yomi* (pembacaan Jepang)

dari sebuah karakter, kemudian yang kedua disebut *on yomi* (pembacaan Cina) dari karakter.

Berikut adalah contoh *kanji* untuk air :

水	雨水	海水	水素	恐水病	水族館	水車
mizu	amamizu	kaisui	suiso	kyōsuibyō	suizokukan	suisha
air	air hujan	air laut	hydrogen	hydrophobia	aquarium	kendaraan air

Bererapa *kanji* telah disederhanakan, seperti contoh berikut :

Tradisional (Cina)	發	賣	龜	齒	傳	讀	關	歸
Disederhanakan (Cina)	发	卖	龟	齿	传	读	关	归
Disederhanakan (Jepang)	発	売	亀	歯	伝	読	関	帰

Ada juga beberapa karakter yang telah dibuat, disebut dengan *kokuji* (karakter nasional).

働く	凪	辷る	匂い	杢	茣蓙	凧
hataraku	nagi	suberu	nioi	moku	goza	tako
bekerja	menyanyi	terpeleset	memcium	tukang kayu	karpet	layangan

2.3 Multimedia

Multimedia adalah suatu kombinasi dari berbagai medium, dimana kombinasi tersebut dapat digunakan untuk tujuanpembelajaran. *Multimedia* juga dapat diartikan sebagai gabungan dari teks, suara, gambar, animasi dan video dengan alat bantu (tool) dan koneksi (link) sehingga pengguna dapat bernavigasi, berinteraksi, berkarya dan berkomunikasi.[5]

Multimedia berasal dari kata multi dan media. Multi berarti banyak, dan media berarti tempat, sarana atau alat yang digunakan untuk menyimpan informasi. Jadi berdasarkan kata "*multimedia*" dapat diasumsikan sebagai wadah atau penyatuan beberapa media yang kemudian didefinisikan sebagai elemen-elemen pembentukan multimedia. Elemen-elemen tersebut berupa teks, gambar, suara, animasi, dan video. *Multimedia* merupakan suatu konsep

dan teknologi baru bidang bidang teknologi informasi, dimana informasi dalam bentuk teks, gambar, suara, animasi, dan video disatukan dalam komputer untuk disimpan, diproses, dan disajikan baik secara linier maupun interaktif.

Menurut Arsyad *multimedia* adalah berbagai macam kombinasi grafik, teks, audio, suara, dan animasi. Penggabungan ini merupakan suatu kesatuan yang secara bersama-sama menampilkan informasi, pesan, atau isi pembelajaran.[5]

Sedangkan Gayeski mengartikan *multimedia* ialah suatu sistem hubungan komunikasi interaktif melalui komputer yang mampu menciptakan, menyimpan, memindahkan, dan mencapai kembali data dan maklumat dalam bentuk teks, grafik, animasi, dan sistem audio.[6]

Penyajian dengan menggabungkan seluruh elemen multimediatersebut menjadikan informasi dalam bentuk multimediayang dapat diterima oleh indera penglihatan dan pendengaran, lebih mendekati bentuk aslinya dalam dunia

sebenarnya. *Multimedia* interaktif adalah bila suatu aplikasi terdapat seluruh elemen multimedia yang ada dan pemakai (*user*) diberi kebebasan/kemampuan untuk mengontrol dan menghidupkan elemen-elemen tersebut.

2.3.1 Komponen Multimedia

Multimedia terdiri dari beberapa komponen, yaitu:

1. Teks

   Teks merupakan dasar dari pengolahan kata dan informasi berbasis multimedia.

2. Grafik

   Secara umum grafik berarti gambar garis (*line drawing*). Manusia sangat berorientasi pada *visual*, dan gambar merupakan sarana yang sangat baik untuk menyajikan informasi.

3. Images

Secara umum *images* berarti gambar *raster* (*halftone drawing*), seperti foto.

4. Animasi

Animasi menggambarkan objek yang bergerak agar kelihatan hidup.

5. Suara

Suara dapat lebih memperjelas pengertian yang ditampilkan dengan cara lain.

2.3.2 Multimedia dalam Pembelajaran

Multimedia mencakup berbagai media yang terintegrasi menjadi satu. Setiap komponen media dapat merangsang satu atau lebih indra manusia. Teori Koehnert mengatakan bahwa semakin banyak indra yang terlibat dalam proses belajar, maka proses belajar tersebut akan menjadi lebih efektif. Secara tegas teori ini menyarankan penggunaan lebih dari satu indera manusia. Oleh karena itu, pemanfaatan *multimedia* dalam pembelajarandapat diharapkan meningkatkan hasil belajar.[6]

Pernyataan di atas berkaitan dengan pendapat Dale yang menyatakan bahwa pemerolehan hasil belajar melalui indra pandang berkisar 75%, melalui indra dengar sekitar 13%, dan melalui indra lainnya sekitar 12%. Hal senada ditegaskan oleh Baugh yang menyatakan bahwa kurang lebih 90% hasil belajar seseorang diperoleh melalui indra pandang, 5% diperoleh melalui indra dengar, dan 5% lagi diperoleh melalui indra lainnya.[7]

Selain itu multimedia juga fleksibel dalam menyesuaikan dengan kecepatan belajar seseorang. Seorang pebelajar yang memiliki kecepatan belajar lebih tinggi dapat lebih cepat menyelesaikan kegiatan belajarnya, sedangkan pebelajar dengan kecepatan belajar lambat dapat menyelesaikan aktivitas belajarnya sesuai dengan kecepatannya masing-masing. Hal ini sangat berbeda dengan pembelajaran di kelas konvensional, dimana setiap

pebelajar dipaksa belajar dengan kecepatan yang ditentukan oleh guru. Pebelajar yang mempunyai kecepatan belajar tinggi dapat merasa bosan, sebaliknya pebelajar dengan kecepatan belajar rendah merasa pembelajaran terlalu cepat untuk diikuti.[5]

2.4 Perangkat *Mobile*

Perangkat *mobile* atau bisa disebut sebagai pernangkat yang mudah dibawa kemana-mana. Perangkat *mobile* menjadi salah satu platform yang paling sering digunakan dan dimanfaatkan dalam berbagai kemajuan di bidang IT. Kelebihan yang berupa *portability* atau mudah dibawa menyebabkan platform ini sangat ideal dalam menerapkan teknologi terkini, misalnya *Computer Vision*.

Perangkat *mobile* yang beredar saat ini memiliki kemampuan *processing* yang cukup baik untuk melakukan proses yang berat seperti *Computer Vision*. Sebagai contohnya adalah perangkat *mobile* yang sering digunakan dalam *Computer Vision* adalah Ponsel pintar (*smartphone*). Ponsel pintar ini dapat melakukan tugas-tugas yang biasa dikerjakan oleh perangkat Desktop. Terdapat dua *platform* ponsel pintar yang mendominasi pasar atau digunakan oleh orang, yaitu Android dan iOS.



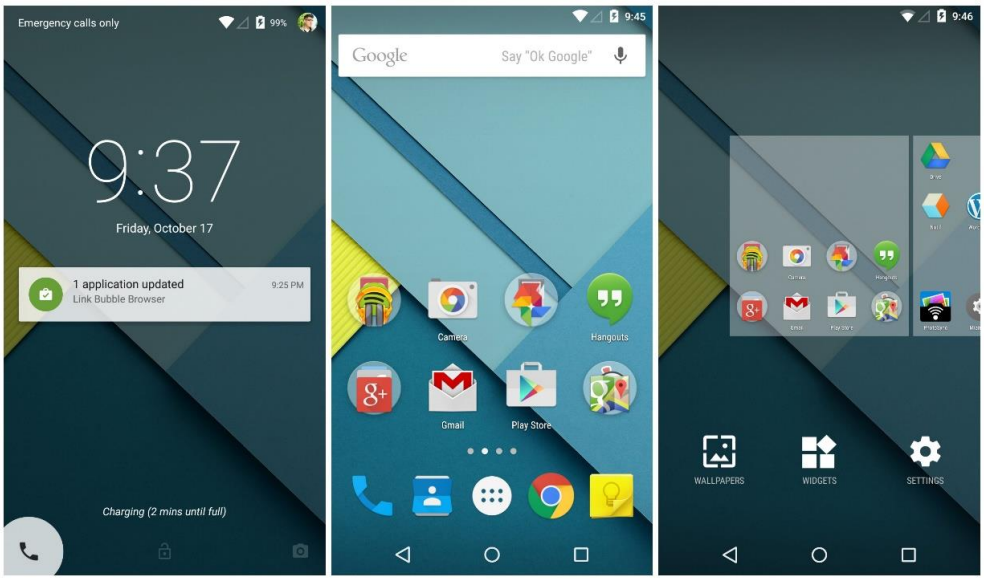
Gambar 2.3 Contoh Perangkat Mobile

2.4.1 Android

Android adalah sistem operasi yang berbasis Linux untuk telepon selulerseperti telepon pintar dan komputer tablet. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

Sistem Operasi Android sendiri mencakup banyak ruang lingkup misalkan *Android 1.5 (Cup Cake) SDK, Android 1.6 (Donut) SDK, Android 2.0/2.1 (Eclair) SDK, Android 2.2 (Froyo – Frozen Yogurt), Android 2.3 (Gingerbread), Android versi 3.0/3.1 (Honeycomb), Android versi 4.0 (ICS: Ice Cream Sandwich), Android versi 4.1 (Jelly Bean), Android versi 4.4 (Kitkat), Android versi 5.0 (Lollipop), Android versi 6.0 (Marshmallow),* dan *Android Developer preview 2 (N).*

Berikut adalah tampilan dari *Android Lollipop* :



Gambar 2.4 *Android Lollipop*

2.5 *Location Based Service* (LBS)

Salah satu ciri mobile phone atau tablet pc adalah portabilitas, sehingga tidak mengherankan bahwa beberapa fitur Android sangat menarik seperti layanan yang memungkinkan anda menemukan, mencari serta menvisualisasikan posisi kita ke dalam peta lokasi fisik seperti *GoogleMaps*. Kita dapat membuat peta berbasis *GoogleMaps* dan menjadikannya sebagai element dalam layout User Interface (UI) yang kita rancang. Kita dapat melakukan akses penuh ke peta *GoogleMaps*, dan memungkinkan kita untuk mengontrol pengaturan tampilan, mengubah tampilan zoom lokasi, dan memindahkan lokasi tampilan.[11]

Location Based Service adalah service yang berfungsi untuk mencari dengan teknologi *Global Positioning Service* (GPS) dan *Google's cell-based location*. Map dan layanan berbasis lokasi menggunakan lintang dan bujur untuk menentukan lokasi geografis, namun sebagai user kita membutuhkan alamat atau posisi *realtime* kita bukan nilai lintang dan bujur. *Android* menyediakan *geocoder* yang mendukung *forward* dan *reverse geocoding*. Menggunakan *geocoder*, anda dapat mengkonversi nilai lintang bujur menjadi alamat dunia nyata atau sebaliknya. [11]

Location based service atau layanan berbasis lokasi adalah istilah umum yang digunakan untuk menggambarkan teknologi yang digunakan untuk menemukan lokasi perangkat yang kita gunakan. Dua unsur utama LBS adalah:

a. *Location Manager* (API Maps)

   Menyediakan *tools*/*resource* untuk LBS, *Application Programming Interface* (API) Maps menyediakan fasilitas untuk menampilkan, memanipulasi maps/peta beserta *feature-feature* lainnya seperti tampilan satelit, *street* (jalan), maupun gabungannya. Paket ini berada pada com.google.android.maps. [11]

b. *Location Providers* (API *Location*)

   Menyediakan teknologi pencarian lokasi yang digunakan oleh *device*/perangkat. API *Location* berhubungan dengan data GPS dan

data lokasi *realtime*. API *Location* berada pada paket Android yaitu dalam paket android.*location*. Dengan *Location Manager*, kita dapat menentukan lokasi kita saat ini, *Track* gerakan/perpindahan, serta kedekatan dengan lokasi tertentu dengan mendeteksi perpindahan. [11]

2.5.1 *Global Positioning System* (GPS)

GPS adalah singkatan dari *Global Positioning System*, yang merupakan sistem navigasi dengan menggunakan teknologi satelit yang dapat menerima sinyal dari satelit. Sistem ini menggunakan 24 satelit yang mengirimkan sinyal gelombang mikro ke bumi. Sinyal ini diterima oleh alat penerima (*receiver*) di permukaan, dimana GPS *receiver* ini akan mengumpulkan informasi dari satelit GPS, seperti:

a. Waktu. GPS *receiver* menerima informasi waktu dari jam atom yang mempunyai keakurasian sangat tinggi.
b. Lokasi. GPS memberikan informasi lokasi dalam tiga dimensi:
    1. Latitude
    2. Longitude
    3. Elevasi
c. Kecepatan. Ketika berpindah tempat, GPS dapat menunjukkan informasi kecepatan berpindah tersebut.
d. Arah perjalanan. GPS dapat menunjukkan arah tujuan.
e. Simpan lokasi. Tempat-tempat yang sudah pernah atau ingin dikunjungi bisa disimpan oleh GPS *receiver*.
f. Komulasi data. GPS *receiver* dapat menyimpan informasi *track*, seperti total perjalanan yang sudah pernah dilakukan, kecepatan rata-rata, kecepatan paling tinggi, kecepatan paling rendah, waktu/jam sampai tujuan, dan sebagainya. [11]

2.5.1.1 Akurasi GPS

Posisi yang ditunjukkan oleh suatu GPS mempunyai faktor kesalahan atau juga disebut tingkat akurasi. Sebagai contoh suatu alat GPS

menunjukkan titik koordinat dengan tingkat akurasi 5 meter, itu berarti posisi pengguna bisa berada dalam range radius 5 meter dari titik yang ditunjukkan tersebut. [11]

Mengapa tingkat akurasi yang terlihat bisa berubah-ubah? Kadang terlihat 10 meter, 15 meter, atau 5 meter. Ada beberapa hal yang mempengaruhi tingkat akurasi tersebut, antara lain:

a. Kesalahan Ephemeris. Terjadi jika satelit tidak dapat mentransmisikan posisinya di orbit dengan tepat.
b. Keadaan Ionosphere. Ionosphere berada pada jarak sekitar 43-50 mil di atas permukaan bumi. Satelit yang melewati ionosphere akan menjadi lambat dikarenakan adanya plasma (gas dengan tingkat kepadatan rendah). Walaupun GPS receiver berusaha untuk mengkoreksi/memperbaiki faktor keterlambatan yang terjadi tetap saja aktivitas tertentu dari plasma bisa menyebabkan kesalahan perhitungan.
c. Keadaan Troposphere. Troposphere adalah bagian terendah dari atmosfer sampai dengan ketinggian sekitar 11 mil dari permukaan tanah. Variasi pada temperatur, tekanan, dan kelembaban bisa menyebabkan perbedaan kecepatan penerimaan gelombang radio.
d. Kesalahan Waktu. Karena penempatan jam atom pada setiap GPS receiver tidak berjalan sebagaimana mestinya. Kesalahan waktu dari GPS receiver yang tidak presisi dapat menimbulkan ketidakakurasian.
e. Kesalahan Multipath. Terjadi karena sinyal satelit membentur permukaan keras (seperti bangunan atau tebing) sebelum mencapai GPS receiver. Hal tersebut bisa menyebabkan terjadinya delay sehingga perhitungan jarak menjadi tidak akurat.
f. Buruknya Sinyal Satelit. Keadaan langit yang terhalang akan menyebabkan GPS sulit menerima data satelit. Sebuah sinyal satelit yang pada hari tertentu diterima dengan sangat bagus belum tentu pada hari lain bisa diterima dengan kualitas yang sama walaupun

user berdiri pada tempat yang sama. Hal tersebut dikarenakan posisi dari satelityang terus bergerak atau bisa juga disebabkan faktor penghalang lain seperti pohon, gedung bertingkat, dan sebagainya.

2.5.1.2 Android dan GPS

Dewasa ini, teknologi berkembang dengan pesat. Dulu ponsel hanya sekedar digunakan untuk menelpon dan SMS saja. Sekarang ponsel sudah menjelma menjadi kotak kecil ajaib yang serba bisa. Salah satunya adalah ponsel dengan sistem operasi Android. [11]

Dengan standarisasi fitur dan hardware yang dimiliki, menjadikan ponsel Android ponsel canggih yang disukai banyak orang. Tidak lagi canggih karena adanya fitur MMS, radio, atau internet berkecepatan tinggi tapi juga karena ditanamkannya fitur teknologi satelit di dalamnya. Ya, perangkat GPS receiver yang dulu besar dan eksklusif, sekarang sudah bisa dimiliki dengan "hanya" membeli sebuah ponsel. [11]

Dengan ponsel berteknologi satelit (GPS), banyak hal yang bisa dilakukan. Ingin melihat di mana posisi user sekarang dalam sebuah peta? Mengambil foto/video yang sudah dilengkapi dengan data koordinat? Ingin tahu jalur olahraga bersepeda yang sudah pernah user lalui? Atau ingin tahu dimana posisi ponsel yang hilang dicuri orang? Semua itu bisa dilakukan dengan ponsel Android yang memiliki fitur GPS. [11]

Tidak hanya itu, user juga dapat pergi ke tempat wisata tertentu dengan dipandu gambar dan suara dari sebuah ponsel! Bahkan lebih jauh lagi, GPS dapat digunakan untuk membantu memberikan peringatan dini terhadap terjadinya bencana alam. [11]

Sekarang ini banyak sekali pengembangpengembang aplikasi untuk sistem operasi Android termasuk aplikasi-aplikasi GPS. Yang menyenangkan aplikasi-aplikasi tersebut jenisnya beragam dan jumlahnya pun banyak. [11]

2.5.1.3 Latitude dan Longitude

Latitude dan longitude adalah sebuah koordinat yang ada di bumi dan digunakan untuk menentukan kedudukan kita di atas bumi.

2.5.1.3.1 Latitude

Latitude adalah garis yang melintang dari kutub utara dan kutub selatan. Titik 0 adalah sudut ekuator, tanda + menunjukan arah ke atas menuju kutub utara, sedangkan tanda minus di koordinat Latitude menuju ke kutub selatan. Titik yang dipakai dari 0 ke 90 derajat arah kutub utara, dan 0 ke -90 derajat ke kutub selatan. [11]

2.5.1.3.2 Longitude

Longitude adalah garis lintang. Angka dari sudut bundar bumi horisontal. Titik diawali dari 0 ke 180 derajat, dan ke arah sebaliknya. Titik 0 dimulai dari garis negara Inggris. Mengarah ke Indonesia akan menjadi angka positif. Kebalikannya koordinat Longitude minus adalah arah kebalikan. [11]

2.6 Tools Pemodelan Aplikasi

Pemodelan dan analisa dapat dianggap sebagai akar konsep dari semua displin teknik karena keutamaannya dalam mendokumentasikan dan mengevaluasi alternatif serta pilihan desain. Pemodelan dilakukan pada awal analisa, spesifikasi dan validasi persyaratan. Persyaratan yang dimaksud adalah semua kebutuhan pengguna, kustomer dan stakeholder yang harus dipenuhi oleh sistem. Dalam pembuatan persyaratan melibatkan :

1. Analisa studi kelayakan dari sistem yang akan dibuat,
2. Memperoleh dan menganalisa kebutuhan dari stakeholder
3. Pembuatan gambaran yang akurat terhadap sistem tentang apa yang harus dilakukan sistem dan apa yang tidak.
4. Validasi terhadap dokumen atau spesifikasi oleh stakeholder

Pemodelan digunakan untuk mengurangi kompleksitas, mendokumentasikan keputusan rancangan dan menfasilitasi komunikasi di dalam tim proyek. Pemodelan bertujuan menyediakan suatu spesifikasi dari sistem yang akan dibangun dengan derajat detail yang cukup bagi implementasi sistem tersebut. Hasil dari proses pemodelan adalah model yang merepresentasikan aspek-aspek yang relevan dari sistem dalam bentuk yang disederhanakan dan juga *comprehensible* (dapat dipahami).

Aplikasi ini dibuat dengan pemodelan berbasis objek, yaitu dengan pemodelan UML (Unified Modeling Language). [8]

2.6.1 *Unified Modelling Language(UML)*

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. *Unified Modeling Language* (UML) adalah himpunan struktur dan teknik untuk pemodelan desain program berorientasi objek (OOP) serta aplikasinya.

UML adalah metodologi untuk mengembangkan sistem OOP dan sekelompok perangkat *tool* untuk mendukung pengembangan sistem tersebut. UML mulai diperkenalkan oleh *Object Management Group*, sebuah organisasi yang telah mengembangkan model, teknologi, dan standar OOP sejak tahun 1980-an. Sekarang UML sudah mulai banyak digunakan oleh para praktisi OOP. UML merupakan dasar bagi perangkat *(tool)* desain berorientasi objek dari IBM [8].

UML merupakan suatu bahasa yang digunakan untuk menentukan, memvisualisasikan, membangun, dan mendokumentasikan suatu sistem informasi.UML dikembangkan sebagai suatu alat untuk analisis dan desain berorientasi objek oleh Grady Booch, Jim Rumbaugh, dan Ivar Jacobson. Namun demikian UML dapat digunakan untuk memahami dan mendokumentasikan setiap sistem informasi. Penggunaan UML dalam industri terus meningkat. Ini merupakan standar terbuka yang menjadikannya

sebagai bahasa pemodelan yang umum dalam industri peranti lunak dan pengembangan sistem [8].

Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan class dan operation dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C [8].

2.6.1.1 Diagram UML

UML menyediakan 4 macam Diagram untuk memodelkan aplikasi berorientasi objek, yaitu:

1. *Use Case Diagram* untuk memodelkan proses bisnis.
2. *Sequence Diagram* untuk memodelkan pengiriman pesan (message) antar objects.
3. *Activity Diagram* untuk memodelkan perilaku *Use Case* dan objects di dalam system.
4. *Class Diagram* untuk memodelkan struktur kelas.

2.6.1.1.1 Usecase Diagram

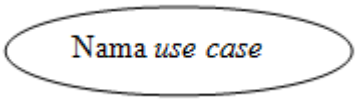
Use case Diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem, create sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. Sebuah use case dapat

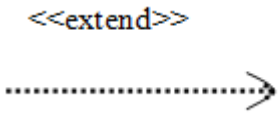
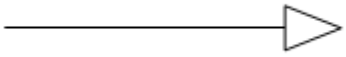
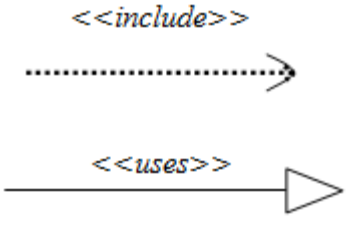
menginklusi fungsionalitas use case lain sebagai bagian dari proses dalam dirinya.

Secara umum diasumsikan bahwa use case yang diinklusi akan dipanggil setiap kali use case yang menginklusi dieksekusi secara normal. Sebuah use case dapat diinklusi oleh lebih dari satu use case lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah use case juga dapat mengekstensi use case lain dengan prilakunyanya sendiri. Sementara hubungan generalisasi antar use case menunjukkan bahwa use case yang satu merupakan spesialisasi dari yang lain [9].

Berikut adalah tabel simbol pada *use case* beserta fungsinya :

Tabel 2.3 Simbol pada *use case* diagram

Simbol	Deskripsi
Use Case 	Fungsional yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja diawal frase nama *use case*.
Aktor / *Actor*  Nama aktor	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat, jadi walaupun simbol aktor adalah gambar orang, belum tentu aktor adalah orang; biasanya menggunakan kata benda di awal frase nama aktor

Simbol	Deskripsi
Asosiasi / *Association*	Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor
Ekstensi / *Extend* 	Relasi *use case* tambahan ke sebuah *use case* yang ditambahkan dapat berdiri sendiri walaupun tanpa *use case* tambahan itu; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.
Generalisasi / *Generalization* 	Hubungan generalisasi dan spesialis (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang umum dari yang lainnya
Menggunakan / *Include / Uses* 	Relasi *use case* tambahan ke sebuah *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini

2.6.1.1.2 *Activity Diagram*

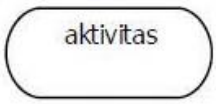
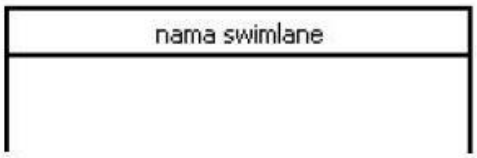
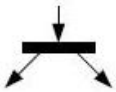
Activity Diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity Diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity Diagram* merupakan *state Diagram* khusus, di mana sebagian besar state adalah action dan sebagian besar transisi *di-trigger* oleh selesainya state sebelumnya (internal processing). Oleh karena itu *activity Diagram* tidak menggambarkan *behaviour internal* sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. *Activity Diagram* dapat dibagi menjadi beberapa object *swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu [8].

Berikut adalah tabel simbol pada *activity diagram* beserta fungsinya :

Tabel 2.4 Simbol pada *activity diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah tatus awal

Simbol	Deskripsi
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan / *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
Penggabungan / *join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas dibangkan menjadi satu
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi
Fork 	Digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel
Join 	Digunakan untuk menunjukkan kegiatan yang digabungkan

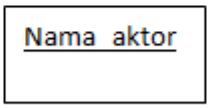
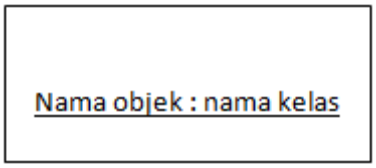
2.6.1.1.3 Sequence Diagram

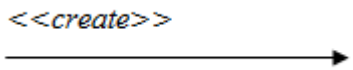
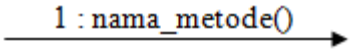
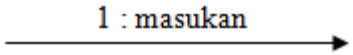
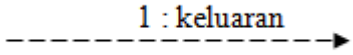
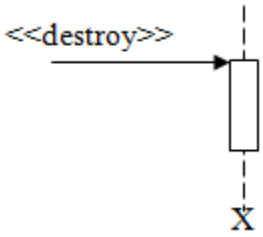
Sequence Diagram menjelaskan secara detil urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari use case: interaksi yang terjadi antar class, operasi apa saja yang terlibat, urutan antar operasi, dan informasi yang diperlukan oleh masing-masing operasi.

Sequence Diagram digunakan khususnya untuk menunjukkan interaksi antara objek dalam urutan sekuensial dimana interaksi itu terjadi. Mirip seperti *class diagram*, para pengembang aplikasi biasanya berfikir bahwa *sequence diagram* adalah eksklusif untuk mereka. Namun, staff dalam organisasi atau perusahaan bisa menemukan bahwa *sequence diagram* berguna untuk menyampaikan bagaimana bisnis yang berjalan saat ini dengan menunjukan berbagai objek bisnis berinteraksi. Selain untuk mendokumentasi urusan organisasi saat ini, sebuah *sequence diagram* berlevel bisnis bisa digunakan sebagai dokumen persyaratan untuk menyampaikan kebutuhan untuk implementasi sistem kedepannya. Selama fase pengumpulan persyaratan, *analyst* bisa mengambil beberapa *use case* ke level selanjutnya dengan menyertakan penambahan atau perbaikan. Ketika itu terjadi, *use case* sering disempurnakan kedalam satu atau lebih *sequence diagram.* [8]

Berikut adalah tabel simbol pada *sequence diagram* beserta fungsinya :

Tabel 2.5 **Simbol pada *Sequence Diagram***

Simbol	Deskripsi
Aktor  Nama_aktor Atau  	Orang, proses atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda diawal frase nama aktor
Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek
Objek  	Menyatakan objek yang berinteraksi pesan
Waktu Aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang berhubungan dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya.

Simbol	Deskripsi
Pesan tipe *create* 	Menyatakan suatu objek membuat objek lain, arah panah mengarah pada objek yang dibuat
Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri
Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainnya
Pesan tipe *return* 	Menyatakan bahwa suatu objek yang sudah menjalankan suatu operasi atau metode menghasilkan kembalian ke objek tertentu.
Pesan tipe *destroy* 	Menyatakan suatu objek mengakhiri hidup objek lain, arah panah mengarah pada objek yang di akhiri.

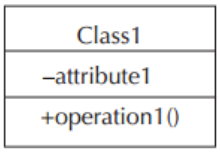
2.6.1.1.4 *Class Diagram*

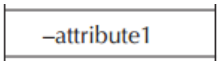
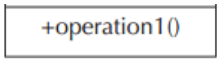
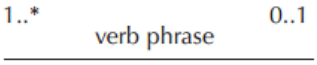
Class Diagram merupakan *Diagram* yang selalu ada di permodelan sistem berorientasi objek. *Class Diagram* menunjukkan hubungan antar class dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan. Sebuah kelas mewakili sekelompok hal yang memiliki keadaan umum dan perilaku. Anda dapat menganggap kelas sebagai blue print untuk sebuah objek dalam suatu

sistem berorientasi objek. Dalam UML, kelas adalah semacam classifier. Misalnya, Volkswagen, Toyota, Ford dan semua mobil, sehingga Anda dapat mewakili mereka menggunakan kelas bernama Mobil. Setiap jenis tertentu mobil adalah sebuah instance dari kelas, atau benda. Sebuah kelas dapat mewakili sebuah konsep yang nyata dan konkret, seperti faktur, dan bisa sesuatu yang abstrak, seperti dokumen atau kendaraan (sebagai lawan faktur, atau sepeda motor lebih dari 1000 cc), atau mungkin mewakili konsep tidak berwujud seperti strategi investasi berisiko tinggi. [8]

Berikut adalah tabel simbol pada *class diagram*:

Tabel 2.6 Simbol pada *class diagram*

Simbol	Deskripsi
Class 	a. Menggambarkan seseorang, tempat, atau sesuatu yang mana sistem membutuhkannya menangkap dan menyimpan informasi b. Mempunyai nama *class* yang ditulis tebal dan ditengah pada baris atas c. Memiliki daftar *attribute* pada baris tengah d. Memiliki daftar *operation* pada baris bawah e. Tidak secara eksplisit menunjukkan operasi yang tersedia untuk semua *class*

Simbol	Deskripsi
Nama atribut / *Attribute name / derived attribut name* 	a. Menunjukkan *properties* yang menggambarkan keadaan sebuah objek b. Bisa berasal dari atribut lainnya, dengan menyertakan tanda *slash* (/) sebelum nama atribut
Operasi / *Operation / Method* 	a. Menunjukkan aksi atau fungsi yang bisa dilakukan *class* b. Bisa digolongkan sebagai sebuah *constructor, query,* atau operasi *update* c. Tanda kurung menunjukkan parameter atau informasi yang dibutuhkan dalam operasi
Asosiasi / *Association* 	a. Menunjukan hubungan atau relasi antar *class* atau sebuah *class* dengan *class* itu sendiri b. Menggunakan label dengan kata kerja yang mewakili hubungan c. Mengandung simbol *Multiplicity*, yang menggambarkan minimum dan maksimum sebuah *class instance* diasosiasikan

Simbol	Deskripsi
	dengan *class instance* yang berhubungan

Dalam *class diagram*, relasi antar *class* dituliskan dengan notasi *multiplicity* yang merepresentasikan jumlah banyaknya objek sebuah *class* yang berelasi dengan sebuah objek lain dari kelas lain yang berasosiasi dengan kelas tersebut. Macam-macam *multiplicity* disajikan dalam **Tabel 2.6** berikut :

Tabel 2.7 *Multiplicity* dan artinya

Multiplicity	Arti
*	Banyak
0	Tepat nol
1	Tepat satu
0..*	Nol atau lebih
1..*	Satu atau lebih

2.7 Tools Pembangun Aplikasi

Programming tool atau Software development tool adalah program komputer yang digunakan developer untuk membuat, men-*debug*, menjaga, dan mendukung program lain dan aplikasi. Istilah ini biasanya merujuk kepada program sederhana, yang bisa dikombinasikan bersama untuk menyelesaikan tugas. Tools dasar yang umum digunakan adalah *source code editor* dan *compiler* atau *interpreter*.

2.7.1 Unity3D

Unity adalah sebuah sebuah tool yang terintegrasi untuk membuat *game*, arsitektur bangunan dan simulasi. *Unity* bisa untuk *games* PC dan *games Online*. Untuk *games Online* memerlukan sebuah plugin, yaitu *Unity Web Player*, sama halnya dengan *Flash Player* pada *Browser*. Bahasa

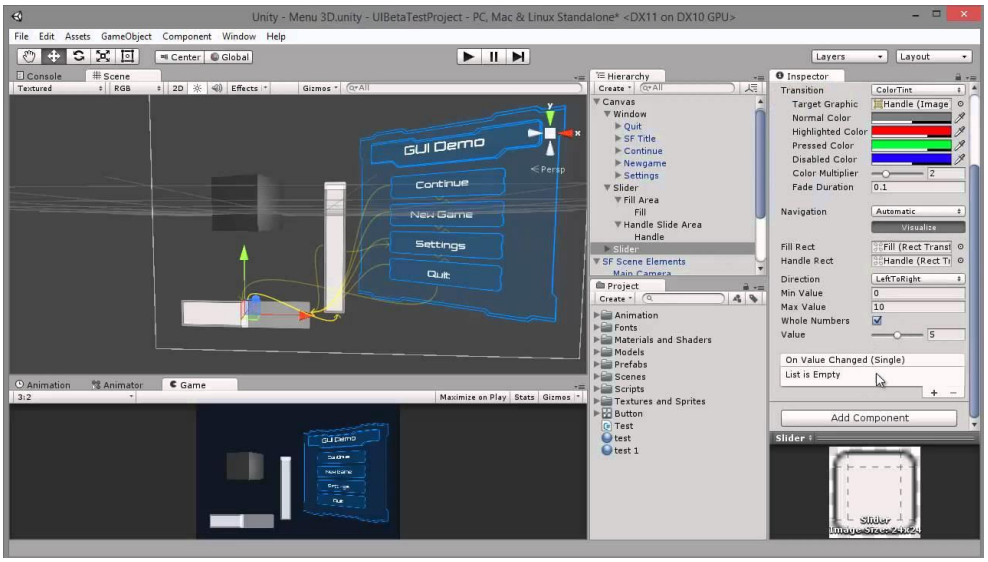
pemrograman yang digunakan bermacam-macam, mulai dari Javascript, C#, dan Boo. [9]

Pada unity, tidak bisa melakukan desain/*modelling*, dikarenakan *unity* bukan tool untuk mendesain. Jadi jika ingin mendesain, harus memerlukan 3D editor lain seperti 3dsmax atau Blender. Banyak hal yang bisa di lakukan di *unity*, ada fitur *audio reverb zone*, *particle effect*, *Sky Box* untuk menambahkan langit, dan masih banyak lagi tentunya, juga bisa langsung mengedit *texture* dari *editor* seperti photoshop dll, unity bagus untuk pemula maupun *expert*.

Keunggulan dari Unity ini antara lain :

1. Mendukung 3 bahasa pemrograman, JavaScript, C#, dan Boo.
2. *Flexible* and *EasyMoving*, *rotating*, dan *scaling objects* hanya perlu sebaris kode. Begitu juga dengan *Duplicating*, *removing*, dan *changing properties*.
3. *Multi Platform Game* bisa di deploy di PC, Mac, Wii, iPhone, iPad dan browser, android.
4. *Visual Properties Variables* yang didefinisikan dengan *scripts* ditampilkan pada *Editor*. Bisa digeser, di-*drag* and *drop*, bisa memilih warna dengan *color picker*.
5. Berbasis.NET –> *Running program* dilakukan dengan Open Source.NET platform, Mono dan dapat di-*build* ke beberapa jenis platform baik desktop maupun mobile.

Tampilan Unity3D seperti pada gambar 2.5;



Gambar 2.5 Tampilan Unity3D

2.7.2 *Application Programming Interface* (API)

API adalah singkatan dari *Application Program Interface*, yakni serangkaian instruksi dan standar pemrograman untuk mengakses aplikasi atau layanan berbasis web. Sebuah perusahaan *software* atau penyedia layanan berbasis web merilis API mereka kepada publik. Dengannya, pengembang lain dapat mendesain aplikasi yang memanfaatkan layanan mereka.

API memungkinkan sebuah aplikasi berkomunikasi dengan aplikasi lain di Internet melalui serangkaian panggilan (*call*). Sebuah API, berdasarkan definisinya, adalah sesuatu yang mendefinisikan cara dua entitas untuk berkomunikasi. Entitas di sini adalah sebuah *software* yang nyata berbeda (dalam layanan) dengan *software* lain.

Dengan API, panggilan-panggilan yang bolak-balik antar aplikasi diatur melalui *web service*. *Web service* adalah kumpulan standar teknis dan protokol, termasuk XML (*Extensible Markup Language*), bahasa umum yang digunakan oleh aplikasi-aplikasi tersebut selama berkomunikasi di Internet.

API sendiri merupakan sekumpulan kode *software* yang ditulis sebagai serangkaian pesan XML. Setiap pesan XML berhubungan dengan fungsi

spesifik dari aplikasi yang akan diajak berkomunikasi. Sebagai contoh, pada API *Facebook*, terdapat pesan XML yang berhubungan dengan fungsi spesifik *wall post*, *wall comment*, *wall like*.

Pengembang aplikasi pihak ketiga menggunakan pesan-pesan XML yang berhubungan dengan fungsi-fungsi spesifik dari layanan web yang akan diajak berkomunikasi. Pengembang bebas memilih fungsi khusus apa saja yang akan diajak berkomunikasi, dan ditampilkan pada aplikasi rancangannya. Sebagai contoh, kita bisa membuat Facebook client yang hanya menampilkan update status teman-teman kita.

Dengan demikian, API adalah standar komunikasi yang dibuka oleh perusahaan software, agar dapat dimanfaatkan oleh pengembang pihak ketiga untuk mendesain aplikasi yang memanfaatkan layanan mereka dengan mudah. [12]

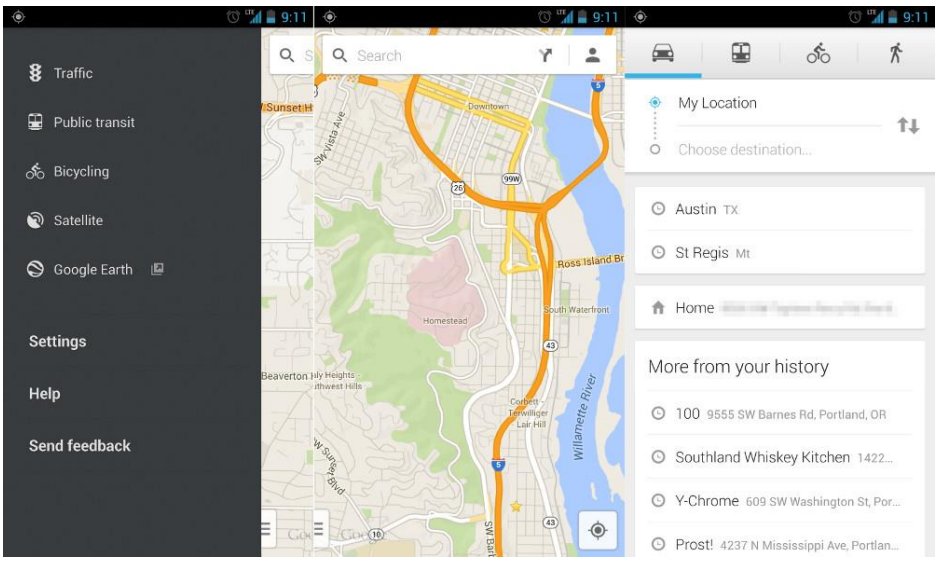
2.7.2.1 *Google Maps* API

Google Maps adalah peta online atau membuka peta secara online, dapat dilakukan secara mudah melalui layanan gratis dari Google. Bahkan layanan ini menyediakan API (Application Programming Interface) yang memungkinkan developer lain untuk memanfaatkan aplikasi ini di aplikasi buatannya. Tampilan *GoogleMaps* pun dapat dipilih, berdasarkan foto asli atau peta gambar rute saja.

GoogleMaps adalah layanan gratis yang diberikan oleh Google dan sangat popular. *GoogleMaps* adalah suatu peta dunia yang dapat kita gunakan untuk melihat suatu daerah seperti pada gambar 2. Dengan kata lain, *GoogleMaps* merupakan suatu peta yang dapat dilihat dengan menggunakan suatu browser. Kita dapat menambahkan fitur *GoogleMaps* dalam web yang telah kita buat atau pada blog kita yang berbayar maupun gratis sekalipun dengan *GoogleMaps* API. *GoogleMaps* API adalah suatu library yang berbentuk *JavaScript*.

Cara membuat *GoogleMaps* untuk ditampilkan pada suatu web atau blog sangat mudah hanya dengan membutuhkan pengetahuan mengenai

HTML serta JavaScript, serta koneksi Internet yang sangat stabil. Dengan menggunakan *GoogleMaps* API, kita dapat menghemat waktu dan biaya untuk membangun aplikasi peta digital yang handal, sehingga kita dapat fokus hanya pada data-data yang akan ditampilkan. Dengan kata lain, kita hanya membuat suatu data sedangkan peta yang akan ditampilkan adalah milik Google sehingga kita tidak dipusingkan dengan mambuat peta suatu lokasi, bahkan dunia. [12]



Gambar 2.6 Tampilan *Google Maps*

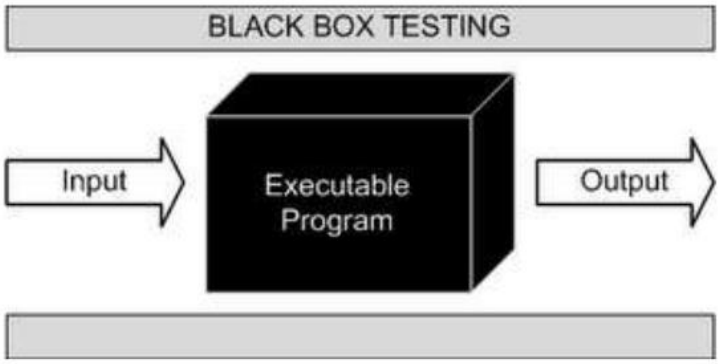
Pada *GoogleMaps* API terdapat 4 jenis pilihan model peta yang disediakan oleh Google, diantaranya adalah:

a. *ROADMAP*, untuk menampilkan peta biasa 2 dimensi.
b. *SATELLITE*, untuk menampilkan foto satelit.
c. *TERRAIN*, untuk menunjukkan relief fisik permukaan bumi dan menunjukkan seberapa tingginya suatu lokasi, contohnya akan menunjukkan gunung dan sungai.
d. *HYBRID*, akan menunjukkan foto satelit yang diatasnya tergambar pula apa yang tampil pada *ROADMAP* (jalan dan nama kota).

2.8 Pengujian Perangkat Lunak (*BlackBox*)

Pengujian perangkat lunak dengan *blackbox* merupakan pendekatan pengujian dimana program dianggap sebagai suatu "*black-box*"(kotak hitam). Program *test case* berbasiskan spesifikasi *test planning* dapat dimulai sejak awal proses pengembangan sistem.

Pengujian *blackbox* berfokus pada persyaratan fungsional perangkat lunak. Metode ini memungkinkan software developer untuk mendapatkan serangkaian kondisi input yang mempergunakan semua persyaratan fungsional program. [12] Gambar 2.7 mengilustrasikan bagaimana pengujian *blackbox*



Gambar 2.7 *Blackbox* testing

Pengujian *blackbox* berusaha menemukan kesalahan dalam kategori :

1. Fungsi-fungsi yang tidak benar atau hilang
2. Kesalahan interface
3. Kesalahan dalam struktur data atau akses database eksternal
4. Kesalahan kinerja
5. Inisialisasi dan kesalahan terminasi

Pengujian *blackbox* diaplikasikan selama tahap akhir pengujian. Pengujian *blackbox* memperhatikan struktur kontrol, sehingga perhatian

berfokus pada domain informasi. Pengujian *blackbox* dirancang untuk dapat menjawab pertanyaan berikut :

1. Bagaimana validitas fungsional diuji?
2. Kelas input apa yang akan membuat test case menjadi baik?
3. Apakah sistem sensitif tehadap nilai input tertentu?
4. Bagaimana batasan dari suatu data diisolasi?
5. Bagaimana kecepatan dan volume data yang dapat ditolelir oleh sistem?
6. Apa pengaruh kombinasi tertentu dari data terhadap operasi sistem