

BAB II

LANDASAN TEORI

2.1. Tinjauan Pustaka

Teknologi yang digunakan dalam penyelesaian proyek akhir ini merupakan pengembangan dari beberapa proyek yang sudah pernah dikerjakan dan dibahas dalam beberapa artikel yang penulis jadikan sebagai referensi acuan. Referensi acuan yang dimaksud antara lain: “*Sistem Keamanan Berbasis Android Vehicle Tracking dengan Mikrokontoler*” (Satrianto, Budiman, & Setiadi, 2016), “*Penggunaan Accelerometer dan Magnetometer pada Sistem Real Time Tracking Indoor Position untuk Studi Kasus pada Gedung Teknik Informatika ITS*” (Mahandhira, Ginardi, & Navastara, 2016), “*Mobile Application for College Bus Tracking*” (Jemilda, Krishnan, Johnson, & Sangeeth, 2015), “*Implementation of Location Base Service on Tourism Places in West Nusa Tenggara by using Smartphone.*” (Gunawan & Purnama, 2015), dan “*Implementation of an Android Application for Location Tracking and Conferencing.*” (Pande & Welekar , 2015).

Dalam jurnal Sistem Keamanan Berbasis Android Vehicle Tracking dengan Mikrokontoler, pelacakan dilakukan menggunakan konsep *Active Tracking*, yakni *tools* yang dibuat hanya digunakan untuk memperoleh data-data yakni koorniat lokasi. *Tools* ini terdiri dari 2 teknologi yang saling bersinambungan yakni Android yang digunakan untuk berinteraksi langsung dengan *user* dan Arduino Uno yang merupakan salah satu *mikrokontroler*, yang dipasang di kendaraan. *User* atau pengguna akan memberikan semacam perintah melalui aplikasi Android miliknya. Jenis perintah itu diantara lain, pelacakan, menyalakan dan mematikan mesin. Pengiriman perintah sesuai

tujuan fungsi yang diinginkan berupa sms dari aplikasi buatan menuju bagian kelistrikan kendaraan. Pengolahan perintah dari Android oleh Arduino pada kendaraan. Bila ada perintah yang tidak dikenal maka perintah itu tidak akan diproses lebih lanjut. Penerimaan perintah dari android oleh arduino untuk melacak atau menyalakan dan mematikan mesin kendaraan.

Dalam jurnal Penggunaan Accelerometer dan Magnetometer pada Sistem Real Time Tracking Indoor Position untuk Studi Kasus pada Gedung Teknik Informatika ITS, aplikasi yang dibuat menggunakan konsep Indoor Positioning System yang dipadukan dengan *magnetometer* dan *accelerometer* yang akan memprediksi pergerakan dari pengguna aplikasi itu. Pendeteksi pergerakan adalah sebuah teknik yang digunakan untuk mendeteksi dan mengetahui apakah suatu pengguna sedang bergerak atau tidak. Teknik ini digunakan dengan cara menghitung nilai signifikansi dari nilai perubahan sensor. Dalam hal ini, pendeteksian arah dan pergerakan pengguna didapatkan dari pembacaan sensor *accelerometer* dan *magnetometer*. *Magnetometer* adalah suatu sistem atau perangkat yang bekerja atas dasar pendeteksian gaya magnet bumi. Biasanya *magnetometer* digunakan untuk menentukan arah mata angin. Pada perangkat *mobile*, *magnetometer* memiliki output berupa besar medan magnet bumi yang diukur dalam tiga sumbu orientasi yaitu x, y, dan z. Dalam hal *positioning*, sensor ini digunakan untuk menentukan arah ketika pengguna sedang menuju suatu tempat. Untuk menghindari kesalahan pengukuran pada keadaan sensor miring, biasanya penggunaan sensor magnetometer akan digabungkan dengan *output* dari *accelerometer*. Karena sensor *magnetometer* digunakan untuk menentukan arah hadap pengguna secara horizontal, maka yang perlu diperhatikan hanyalah perubahan nilai pada orientasi sumbu x saja.

Penggunaan sensor *accelerometer* pada sistem ini adalah untuk mengetahui apakah pengguna sedang bergerak maju atau tidak. Perlu diperhatikan bahwa sistem ini selalu menganggap segala pergerakan adalah maju ke depan. Oleh karena itu, yang perlu diperhatikan cukup nilai dari sumbu *y* apakah melebihi *threshold* yang ditentukan atau tidak. Jika nilai sumbu *y* tidak melebihi *threshold*, maka pengguna dianggap sedang berjalan. Kemudian marker yang menandakan posisi pengguna saat itu akan digeser dengan jarak tertentu ke arah yang telah ditentukan sensor *magnetometer*. Saat pengguna berjalan menggunakan aplikasi, posisi *smartphone* harus berada dalam kondisi *steady hand*, yaitu *smartphone* selalu berada dalam posisi mendatar. Hal ini dikarenakan *accelerometer* akan memprediksi pergerakan maju pengguna, sehingga jika posisi *smartphone* tidak selalu mendatar, akan mempengaruhi nilai sumbu *y* yang berubah secara tidak normal. Perubahan nilai sumbu *y* yang tidak normal ini nantinya juga akan mempengaruhi step detection pada sistem, dimana jika guncangan atau perubahan nilai pada sumbu *y* melebihi *threshold* tidak akan dianggap sebagai sebuah langkah. Hal ini mengakibatkan sistem tidak akan menunjukkan performa yang bagus.

Dalam jurnal *Mobile Application for College Bus Tracking*, konsep dari *location-based service* digunakan untuk melacak bus yang beroperasi. Ada tiga pihak yang menggunakan sistem ini, yakni *admin*, supir dan pengguna. *Admin* bisa login ke akun *admin* setelah autentikasi dan otorisasi. Dia bisa masukkan rincian rute baru dan juga dia dapat memilih rute dari daftar rute dan kemudian pemberhentian yang sesuai ditampilkan. Dia memiliki pilihan untuk menambahkan atau menghapus rute. Dia juga memiliki pilihan untuk memodifikasi atau menghapus pemberhentian dari rute. Jika *admin* ingin mengatakan informasi apapun ke pengemudi maka dia bisa mengirim

pesan ke mobil pengemudi via *browser*. Dia juga bisa memasukkan rincian siswa baru dan bisa melihat daftar siswa. Dia memiliki pilihan untuk menambahkan atau menghapus rincian siswa.

Klien harus memasukkan nomor register dan nomor ponsel untuk login aplikasi. Untuk mencari bus, klien harus memasukkan nomor bus di bilah pencarian. Kemudian peta ditampilkan yang menunjukkan lokasi bus saat ini. Dia juga bisa menerima pemberitahuan peringatan saat bus sampai di pemberhentian terdekat. Saat aplikasi diluncurkan, aktivitas Rumah mengambil rute dari server dan mengikatnya ke pemintal agar klien dapat memilihnya. Ketika klien memilih rute, berhenti yang sesuai diambil dari server dan diikat ke pemintal agar klien dapat memilihnya. Jika klien memilih "*Get Location*" maka rincian lokasi bus untuk rute tersebut diambil. Jika klien memilih "*Show Map*" maka lokasi bus pada peta akan ditampilkan.

Driver harus memasukkan nomor bus, nomor rute, dan nomor ponsel untuk login aplikasi. Pekerjaan pengemudi hanya start dan stop bus. Saat aplikasi diluncurkan, *homeActivity* mengambil rute dari server dan mengikatnya ke pemintal agar pengemudi bisa memilihnya. Jika pengemudi memilih "Start", lokasi bus akan diunggah ke *server*. Jika pengemudi memilih "Stop" maka pengunggahan lokasi bus dihentikan.

Dalam jurnal *Implementation of Location Base Service on Tourism Places in West Nusa Tenggara by using Smartphone*, konsep yang digunakan adalah *passive tracking* yakni, alat atau aplikasi hanya digunakan untuk menyimpan data – data lokasi saja. Aplikasi dimulai dengan menampilkan *splash screen* selama 5 detik untuk memberikan aplikasi *time data collection* ke server, setelah itu akan masuk ke menu utama. Selanjutnya pada halaman utama terdapat 4 menu yaitu tujuan wisata, paket perjalanan, artikel dan tentang. Saat pengguna memilih menu tujuan wisata, maka aplikasi akan menampilkan informasi rinci

tentang tujuan wisata berupa gambar dan penjelasan rinci tentang objek wisata. Di halaman detail informasi terdapat tiga menu yaitu obyek wisata, galeri foto dan share travel information. Objek lokasi hanya menampilkan koordinat tempat wisata di Google Maps, jika koordinat yang kita pilih aplikasi akan menampilkan jalan menuju navigasi atau dalam bentuk tempat wisata mulai dari Google Maps hingga posisi pengguna mengkoordinasikan tempat wisata.

Dalam jurnal *Implementation of an Android Application for Location Tracking and Conferencing*, sistem yang dibuat menggunakan Android sebagai dasar *platform*-nya dan menggunakan A-GPS untuk pelacakannya. Sistem yang diusulkan adalah sistem yang akan menyediakan layanan berbasis lokasi pelacakan lokasi. Pelacakan lokasi mengacu pada pencapaian. Posisi saat objek diam atau bergerak. Aplikasi ini akan memungkinkan karyawan bekerja di lapangan untuk mencari dan melacak karyawan lain yang berada di dekatnya sehingga mereka dapat saling membantu saat berada dalam masalah. Lokasi karyawan juga akan diketahui oleh administrator. Karyawan dapat menemukan lokasi pegawai terdekat. Karyawan juga akan diizinkan melakukan obrolan video dengan karyawan lain atau dengan administrator secara langsung. Obrolan video adalah alat yang fantastis dalam situasi yang tepat. Komunikasi melalui media visual seringkali lebih jelas daripada komunikasi hanya melalui teks atau audio. *Server* akan menjaga data yang relevan dari obrolan karyawan dan video akan terjadi dengan menggunakan internet.

2.2. Landasan Teori

2.2.1. Sistem

Secara sederhana, suatu sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel

yang terorganisir, saling berinteraksi, saling tergantung satu sama lain, dan terpadu.

Sebuah sistem mempunyai karakteristik atau sifat-sifat tertentu yang mencirikan bahwa hal tersebut bisa dikatakan sebagai suatu sistem. Adapun karakteristik yang dimaksud adalah sebagai berikut:

1) Komponen Sistem (Components System)

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, artinya saling bekerja sama membentuk satu kesatuan. Komponen-komponen sistem tersebut dapat berupa suatu subsistem. Setiap subsistem memiliki sifat dari sistem yang menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan.

2) Batas Sistem (Boundary System)

Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem yang lain atau sistem dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan yang tidak dapat dipisahkan.

3) Lingkungan Luar Sistem (Environment System)

Bentuk apapun yang ada di luar ruang lingkup atau batasan sistem yang mempengaruhi operasi sistem tersebut disebut lingkungan luar sistem. Lingkungan luar sistem ini dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut. Dengan demikian, lingkungan luar tersebut harus tetap dijaga dan dipelihara. Lingkungan luar yang merugikan harus dikendalikan. Kalau tidak, maka akan mengganggu kelangsungan hidup dari sistem tersebut.

4) Penghubung Sistem (Interface System)

Media yang menghubungkan sistem dengan subsistem lain disebut penghubung sistem. Penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lain. Bentuk keluaran dari satu subsistem akan menjadi masukan untuk subsistem lain melalui penghubung tersebut. Dengan demikian, dapat terjadi suatu integrasi sistem yang membentuk satu kesatuan.

5) Masukan Sistem (Input System)

Energi yang dimasukkan ke dalam sistem disebut masukan sistem, yang dapat berupa pemeliharaan (maintenance input) dan sinyal (signal input). Maintenance input adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. Signal input adalah energi yang diproses untuk mendapatkan keluaran. Contoh, di dalam suatu unit sistem komputer, program adalah maintenance input yang digunakan untuk mengoperasikan komputernya dan data adalah signal input untuk diolah menjadi informasi.

6) Pengolahan Sistem (Processing System)

Suatu sistem dapat mempunyai suatu proses yang akan mengubah masukan menjadi keluaran, contohnya adalah sistem akuntansi. Sistem ini akan mengolah data transaksi menjadi laporan-laporan yang dibutuhkan oleh pihak manajemen.

7) Keluaran Sistem (Output System)

Hasil energi diolah dan diklasifikasikan menjadi keluaran yang berguna. Keluaran ini merupakan masukan bagi subsistem yang lain seperti sistem informasi. Keluaran yang dihasilkan adalah informasi. Informasi ini dapat digunakan sebagai masukan untuk

pengambilan keputusan atau hal-hal lain yang menjadi input bagi subsistem lain.

8) Sasaran Sistem (Objective) dan tujuan (Goals)

Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat deterministic. Kalau suatu sistem tidak memiliki sasaran maka operasi sistem tidak ada gunanya. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuan yang telah direncanakan. (Sutabri, 2012)

2.2.2. *Flowchat*





Flowchart (Diagram Alir) adalah bagan (*Chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur sistem secara logika. *Flowchart* merupakan metode untuk menggambarkan tahap-tahap pemecahan masalah dengan merepresentasikan simbol-simbol tertentu yang mudah dimengerti, mudah digunakan dan standar. Tujuan penggunaan flowchart adalah untuk menggambarkan suatu tahapan penyelesaian masalah secara sederhana, terurai, rapi, dan jelas dengan menggunakan simbol-simbol yang standar. Tahapan penyelesaian masalah yang disajikan harus jelas, sederhana, dan tepat. (Jogiyanto, 2005)

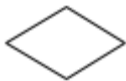



Pada waktu akan menggambar suatu bagan alir, programmer dapat mengikuti pedoman-pedoman sebagai berikut : (Jogiyanto, 2005)

1. Bagan alir sebaiknya digambar dari atas ke bawah dan mulai dari bagian kiri suatu halaman.
2. Kegiatan di dalam bagan alir harus ditunjukkan dengan jelas.

3. Harus ditunjukkan dari mana kegiatan akan dimulai dan di mana akan berakhirnya.
4. Masing-masing kegiatan di dalam bagan alir sebaiknya digunakan kata yang mewakili suatu pekerjaan.
5. Masing-masing kegiatan di dalam bagan alir harus di dalam urutan yang semestinya.
6. Kegiatan yang terpotong dan akan disambung di tempat lain harus ditunjukkan dengan jelas menggunakan simbol penghubung.
7. Gunakan simbol-simbol bagan alir standar.

Tabel 2.1 Tabel Simbol *Flowchart* (Jogiyanto, 2005)

Simbol	Keterangan
	Penghubung Simbol untuk keluar/masuk atau proses dalam lembar atau halaman lain.
	Input Output Simbol yang menyatakan proses <i>input</i> dan <i>output</i> tanpa tergantung dengan jenis peralatannya.
	Dokumen Simbol yang menyatakan input berasal dari dokumen dalam bentuk kertas atau ouput dicetak di kertas.
	Simbol Manual Simbol yang menunjukan pengolahan yang tidak dilakukan oleh komputer.

	Kondisi Simbol keputusan yang menunjukkan kondisi.
	Terminal Simbol yang menunjukan untuk permulaan atau akhir suatu
	Proses Simbol yang menunjukan pengolahan dilakukan oleh komputer.
	Penghubung Simbol untuk keluar/masuk prosedur atau proses dalam lembar atau halaman yang masih sama.

2.2.3. *Unified Modeling Language*



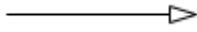


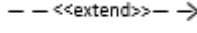
Unified Modeling Language (UML) adalah sebuah bahasa pemodelan visual yang dirancang khusus untuk pengembangan dan analisis sistem berorientasi objek dan desain. UML pertama kali dikembangkan oleh Grady Booch, Jim Rumbaugh, dan Ivars Jacobson pada pertengahan tahun 1990. (Siau & Cao, 2001)

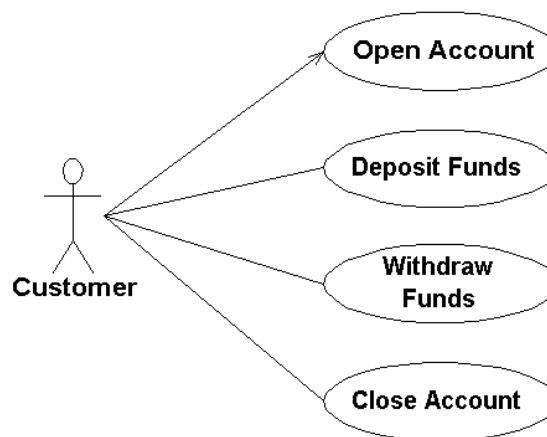
UML menyediakan 4 macam diagram untuk memodelkan aplikasi perangkat lunak berorientasi objek yaitu: (Ambler, 2005)

1. *Use Case Diagram*

Use Case Diagram adalah sebuah diagram yang menunjukkan hubungan antara actors dan use cases. Digunakan untuk analisis dan desain sebuah sistem.

Tabel 2.2 Tabel Simbol *Use Case Diagram*

Simbol	Keterangan
	Actor Menggambarkan seseorang yang berinteraksi dengan sistem, di mana hanya bisa menginputkan informasi dan menerima informasi dari sistem dan tidak memegang kendali pada <i>use case</i> .
	Use Case Gambaran fungsional sistem yang akan di buat, agar pengguna lebih mengerti penggunaan sistem.
	Association Menghubungkan link antar element.
	Generalization Sebuah elemen yang menjadi spesialisasi dari elemen yang lain.
	Include Kelakuan yang harus terpenuhi agar sebuah event dapat terjadi, dimana pada kondisi ini sebuah use case adalah bagian dari use case lainnya.
	Extend Kelakuan yang hanya berjalan di bawah kondisi tertentu seperti menggerakkan alarm.





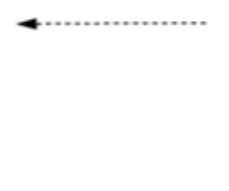
Gambar 2.1 Contoh *Use Case Diagram*

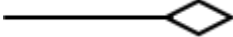
2. *Class Diagram*

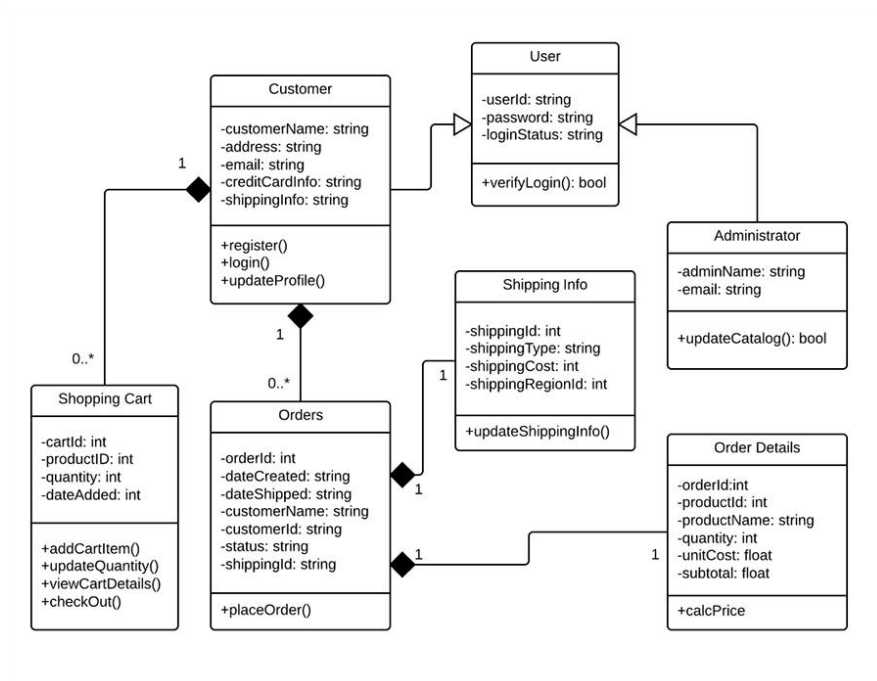
Class Diagram adalah sebuah diagram yang menunjukkan hubungan antar *class* yang didalamnya terdapat atribut dan fungsi dari suatu objek.

Tabel 2.3 Tabel Simbol *Class Diagram*

Simbol	Keterangan
	<p>Class</p> <p><i>Class</i> adalah blok - blok pembangun pada pemrograman berorientasi obyek. Sebuah <i>class</i> digambarkan sebagai sebuah kotak yang terbagi atas 3 bagian. Bagian atas adalah bagian nama dari <i>class</i>. Bagian tengah mendefinisikan <i>property</i> atau atribut <i>class</i>. Bagian akhir</p>

	mendefinisikan <i>method-method</i> dari sebuah <i>class</i> .
	Association Sebuah asosiasi merupakan sebuah relationship paling umum antara 2 <i>class</i> dan dilambangkan oleh sebuah garis yang menghubungkan antara 2 <i>class</i> . Garis ini bisa melambangkan tipe-tipe relationship dan juga dapat menampilkan hukum-hukum multiplisitas pada sebuah <i>relationship</i> .(Contoh: <i>One-to-one</i> , <i>one-to-many</i> , <i>many-to-many</i>)
	Composition Jika sebuah <i>class</i> tidak bisa berdiri sendiri dan harus merupakan bagian dari <i>class</i> yang lain, maka <i>class</i> tersebut memiliki relasi <i>Composition</i> terhadap <i>class</i> tempat dia bergantung tersebut. Sebuah <i>relationship composition</i> digambarkan sebagai garis dengan ujung berbentuk jajaran genjang berisi/solid.
	Dependency Kadangkala sebuah <i>class</i> menggunakan <i>class</i> yang lain. Hal ini disebut <i>dependency</i> . Umumnya

	<p>penggunaan <i>dependency</i> digunakan untuk menunjukkan operasi pada suatu <i>class</i> yang menggunakan <i>class</i> yang lain. Sebuah <i>dependency</i> dilambangkan sebagai sebuah panah bertitik-titik.</p>
	<p>Aggregation</p> <p><i>Aggregation</i> mengindikasikan keseluruhan bagian relationship dan biasanya disebut sebagai relasi.</p>

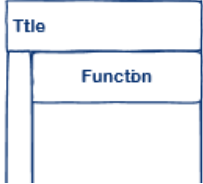






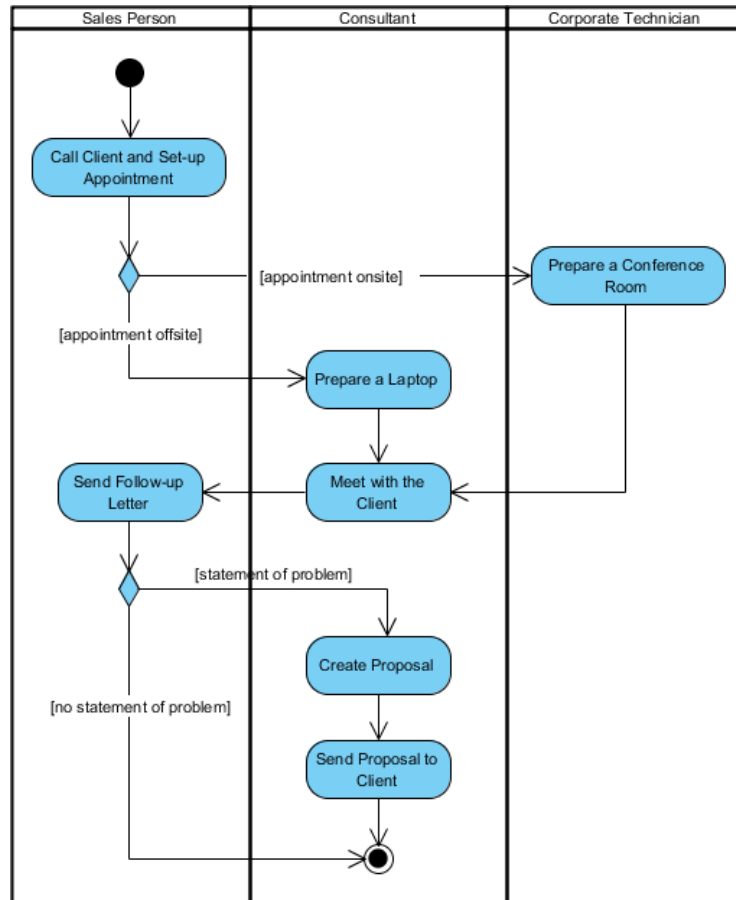
Gambar 2.2 Contoh *Class Diagram*

3. Activity Diagram

Activity diagram adalah suatu diagram yang menggambarkan konsep aliran data/kontrol, aksi terstruktur serta dirancang dengan baik dalam suatu sistem.

Tabel 2.4 Tabel Simbol *Activity Diagram*

Simbol	Keterangan
	Swimlane Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi
	Aktivitas Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
	Percabangan / Decision Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
	Initial Node Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
	Final Node Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir








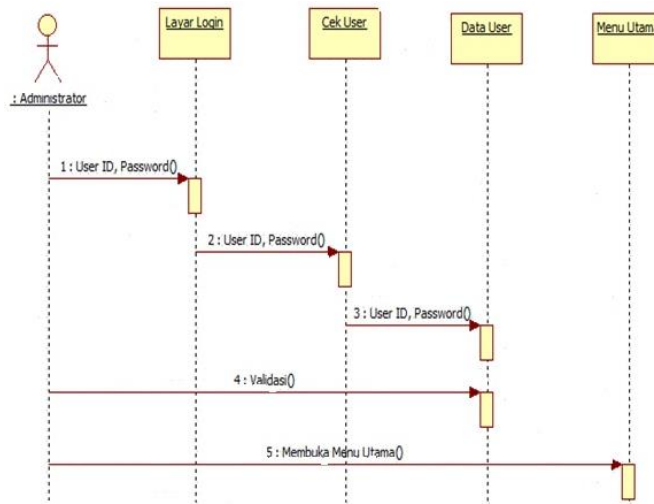
Gambar 2.3 Contoh *Activity Diagram*

4. *Sequence Diagram*

Sequence Diagram adalah sebuah diagram yang menggambarkan kolaborasi dari objek-objek yang saling berinteraksi antar elemen dari suatu *class*.

Tabel 2.5 Tabel Simbol *Sequence Diagram*

Simbol	Keterangan
	Garis Hidup / Lifeline Menyatakan kehidupan suatu objek
	Objek Menyatakan objek yang berinteraksi pesan
	Waktu Aktif Menyatakan objek dalam keadaan aktif dan berinteraksi pesan
	Pesan Tipe <i>Send</i> Menyatakan bahwa satu objek mengirimkan data/masukan/informasi ke objek lainnya, arah panah mengarah pada objek yang dikirim
	Pesan Tipe <i>Return</i> Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian



Gambar 2.4 Contoh *Sequence Diagram*

2.2.4. Location Base Services

2.2.4.1. Definisi

LBS (Location Base Service) adalah sebuah layanan berbasis lokasi, yaitu sebuah layanan berbasis internet yang mampu menampilkan posisi secara geografis dari perangkat bergerak atau ponsel yang kita bawa, atau memberi informasi lokasi dari alamat yang kita inginkan.

Selain itu, LBS tersebut terdiri dari beberapa komponen di antaranya *mobile devices*, *communication network*, *position component*, dan *service and content provider*. *Mobile devices* merupakan komponen yang sangat penting. Piranti *mobile* tersebut diantaranya adalah smartphone, PDA, dan lainnya yang dapat berfungsi sebagai alat navigasi atau seperti halnya alat navigasi berbasis GPS.

Komponen *communication network* ini berupa jaringan telekomunikasi bergerak yang memindahkan data pengguna dari perangkat ke penyedia layanan. *Position component* yang

dimaksud adalah posisi pengguna harus ditentukan. Posisi ini dapat didapatkan dengan jaringan telekomunikasi atau dengan GPS. Sedangkan *service and content provider* adalah penyedia layanan yang menyediakan layanan berbeda ke pengguna seperti pencarian rute, kalkulasi posisi, dan lainnya.

2.2.4.2. Metode

Sebetulnya metode LBS hanya terdiri dari 2 metode. Penentuan metode berdasarkan layanan aplikasi yang digunakan oleh ponsel masing-masing. Berikut sedikit keterangan tentang metode LBS.

1. **Device based (Location base service berdasarkan alat.)** Metode ini berdasarkan alat GPS (Global Positioning System) yang ada di ponsel atau device lainnya. Ponsel mengandalkan kemampuan dari device GPS.

2. **Network Base (Berdasarkan Jaringan).** Terdapat beberapa metode pada network base ini:

- a. **Cell of Origin**

Metode ini adalah metode yang paling sederhana. Bekerja berdasarkan *cell* jaringan yang ada, di mana masing-masing *cell* dijangkau oleh satu BTS (*Base Tower Station*). Lokasi ditentukan berdasarkan letak dari ponsel yang berada pada *cell*. Jadi penentuan Lokasi dari ponsel akan terdeteksi dari BTS yang menjangkaunya. Metode ini adalah metode paling sederhana, namun mempunyai tingkat akurasi yang

rendah, sebab hanya mengandalkan satu BTS dan ponsel untuk mendapatkan sudut lokasi.

b. *Angle of Arrival (AOA)*

Metode ini membutuhkan minimal tiga BTS. Penentuan lokasi didapatkan dari sudut (*angle*) yang dibentuk oleh Ponsel dengan BTS-BTS yang menjangkanya. Sudut ini biasa disebut Triangulasi (segitiga). Tiga BTS akan membentuk segitiga, dan setiap BTS akan mendeteksi lokasi ponsel berdasarkan garis sudutnya. Metode ini lebih akurat dibanding *Cell of Origin*.

c. *Time difference of Arrival (TDOA)*

Metode ini berdasarkan pada perbedaan waktu kedatangan. Jadi informasi lokasi dihasilkan dari perpotongan jarak bujur lingkaran yang dibentuk atas perbedaan waktu kedatangan yang dihasilkan. Metode ini memiliki tingkat akurasi yang lebih baik.

d. *Enhanced Observed Time Difference (TDOA)*

Metode ini adalah pengembangan dari metode *Difference Of Arrival (TDOA)*. Berbedanya adalah Tiga BTS akan selalu mengamati perbedaan sinyal waktu perjalanan ponsel yaitu dengan cara mentransmisikan sinyal ke ponsel tersebut. Metode ini membutuhkan perangkat tambahan pada ponsel.

e. *Location Pattern Matching (LPM)*

Metode ini berdasarkan lokasi dari ponsel. Cara kerjanya adalah BTS menganalisis lokasi ponsel tersebut dari pantulan gelombang elektromagnetik yang dihasilkan oleh ponsel terhadap kondisi lokasi daerah. Hasil analisis berupa tekstur daerah. Lalu BTS

membandingkan tekstur daerah tersebut dengan database yang sudah ada. *Database* ini berisi dari tekstur area. Metode ini mengandalkan dari database lokasi dari BTS-BTS.

f. *Hybrid Method*

Metode ini mengkombinasikan antara metode *device based* (GPS) dengan *Network Based*. Pada proses pemanfaatannya, setiap vendor smartphone mempunyai wewenang masing-masing untuk memilih metode mana yang digunakan.

2.2.5. Android

2.2.5.1. Pengertian Android

Android adalah sistem operasi untuk perangkat mobile yang memiliki basis Linux yang mencakup seluruh sistem operasi, *middleware*, dan aplikasi. Android menyediakan sebuah *platform* yang terbuka (*open source*) bagi para pengembang / *developer* untuk dapat membuat aplikasi mereka sesuai dengan kreativitas mereka sendiri yang digunakan untuk perangkat lunak *mobile*, diantaranya *handphone*, *smartphone*, dan tablet. (Supardi, 2011). Pada awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat *software* yang diperuntukkan 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 (Nugroho, 2014)

Android merupakan *platform* mobile yang memiliki tiga hal sebagai berikut: (Syafaat, 2011)

1. Lengkap (*complete platform*), karena Android menyediakan banyak *tools* dalam membuat dan mengembangkan aplikasi.
2. Terbuka (*open source platform*), karena pengembang secara bebas dapat mengembangkan aplikasi dan *platform* pada Android dan disediakan melalui lisensi *open source*.
3. Free (*free platform*), karena Android merupakan *platform open source* untuk dikembangkan dan tidak ada *royalty* dalam pengembangan aplikasi pada Android.

2.2.5.2. Versi Android

Adapun versi Android yang pernah dirilis adalah sebagai berikut: (Google.Inc, 2017)

1. Android 1.0 (API level 1) dirilis pada 23 September 2008. (Android Police, 2013)
2. Android 1.1 (Secara internal dikenal sebagai “Petit Four”) (API level 2) dirilis pada 09 Februari 2009. (Android Police, 2013)
3. Android 1.5 Cupcake (API level 3, NDK 1) dirilis pada 30 April 2009 (Google.Inc, 2017)
4. Android 1.6 Donut (API level 4, NDK 2) dirilis pada 15 September 2009. (Google.Inc, 2017)
5. Android 2.0 Eclair (API level 5) dirilis pada 26 Oktober 2009. (Google.Inc, 2017)
6. Android 2.0.1 Eclair (API level 6) dirilis pada 03 Desember 2009. (Google.Inc, 2017)
7. Android 2.1 Eclair (API level 7, NDK 3) dirilis pada 12 Januari 2010. (Google.Inc, 2017)
8. Android 2.2.x Froyo (API level 8, NDK 4) dirilis pada 20 Mei 2010, 18 Januari 2011, 22 Januari 2011, 21 November 2011. (Google.Inc, 2017)

9. Android 2.3 s/d 2.3.2 Gingerbread (API level 9, NDK 5) dirilis pada 06 Desember 2010, Desember 2010, Januari 2011. (Google.Inc, 2017)
10. Android 2.3.3 s/d 2.3.7 Gingerbread (API level 10) dirilis pada 09 Februari 2011, 28 April 2011, 25 Juli 2011, 02 September 2011, 21 September 2011. (Google.Inc, 2017)
11. Android 3.0 Honeycomb (API level 11) dirilis pada 22 Februari 2011. (Google.Inc, 2017)
12. Android 3.1 Honeycomb (API level 12, NDK 6) dirilis pada 10 Mei 2011. (Google.Inc, 2017)
13. Android 3.2.x Honeycomb (API level 13) dirilis pada 15 Juli 2011, 20 September 2011, 30 Agustus 2011, Desember 2011, Januari 2012, Februari 2012. (Google.Inc, 2017)
14. Android 4.0.1 s/d 4.0.2 Ice Cream Sandwich (API level 14, NDK 7) dirilis pada 19 Oktober 2011, 21 Oktober 2011, 28 November 2011. (Google.Inc, 2017)
15. Android 4.0.3 s/d 4.0.4 Ice Cream Sandwich (API level 15, NDK 8) dirilis pada 16 Desember 2011, 29 Maret 2012. (Google.Inc, 2017)
16. Android 4.1.x Jelly Bean (API level 16) dirilis pada 9 Juli 2012, 23 Juli 2012, 09 Oktober 2012. (Google.Inc, 2017)
17. Android 4.2.x Jelly Bean (API level 17) dirilis pada 13 November 2012, 27 November 2012, 11 Februari 2013. (Google.Inc, 2017)
18. Android 4.3.x Jelly Bean (API level 18) dirilis pada 24 Juli 2013. (Google.Inc, 2017)
19. Android 4.4 KitKat (API level 19) dirilis pada 31 Oktober 2013. (Google.Inc, 2017)
20. Android 4.4.4 KitKat (API level 20) dirilis pada 19 Juni 2013. (Google.Inc, 2017)

21. Android 5.0 Lollipop (API level 21) dirilis pada 12 November 2014. (Google.Inc, 2017)
22. Android 5.1 Lollipop (API level 22) dirilis pada 9 Maret 2015. (Google.Inc, 2017)
23. Android 6.0 Marshmallow (API level 23) dirilis pada 12 Mei 2015. (Google.Inc, 2017)
24. Android 7.0 Nougat (API level 24) dirilis pada 22 Agustus 2016. (Google.Inc, 2017)
25. Android 7.1 Nougat (API level 25) dirilis pada 4 Oktober 2016. (Google.Inc, 2017)
26. Android 8.0.0 Oreo (API level 26) dirilis pada 21 Agustus 2017. (Google.Inc, 2017)

2.2.6. Google Maps

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.

Google Maps adalah layanan gratis yang diberikan oleh Google dan sangat populer. *Google Maps* adalah suatu peta dunia yang dapat kita gunakan untuk melihat suatu daerah. Dengan kata lain, *Google Maps* merupakan suatu peta yang dapat dilihat dengan menggunakan suatu browser.

Kita dapat menambahkan fitur *Google Maps* dalam web yang telah kita buat atau pada blog kita yang berbayar maupun gratis sekalipun dengan *Google Maps API*. *Google Maps API* adalah

suatu library yang berbentuk *JavaScript*. Cara membuat *Google Maps* 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 *Google Maps 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 membuat peta suatu lokasi, bahkan dunia.

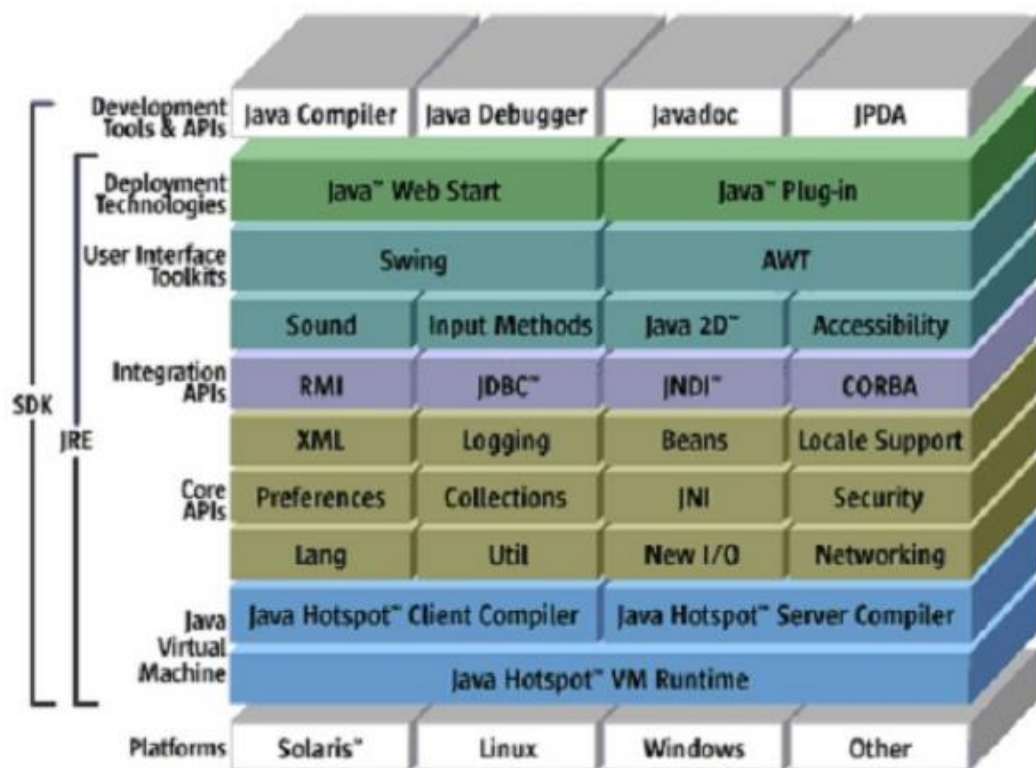
Pada *Google Maps API* terdapat 4 jenis pilihan model peta yang disediakan oleh Google, diantaranya adalah:

- 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). (Rompas, Sinsuw, Sompie, & Lumenta, 2012)

2.2.7. Java

Java adalah perangkat lunak produksi Sun Microsystem Inc., yang merupakan perangkat lunak yang bertujuan mudah dipelajari. Aplikasi-aplikasi yang dapat di buat dengan java, meliputi program web, program desktop, program handphone. Saat ini java diakuisisi oleh Oracle Imc.. (Supardi, 2011)

Bahasa pemrograman Java disebut juga dengan bahasa pemrograman tingkat tinggi dikarenakan struktur bahasanya dapat dikatakan hampir sama dengan bahasa manusia pada umumnya, sehingga programmer tidak terlalu susah beradaptasi dan dapat menyelesaikan permasalahan pada saat melakukan proses coding. Selain itu kepopuleran ini dikarenakan banyaknya keunggulan-keunggulan yang diberikan diantaranya adalah dapat berjalan di berbagai jenis platform, sehingga programmer dalam membuat sebuah aplikasi pada satu jenis platform, dapat digunakan pada berbagai jenis platform. (Martono, 2011). Arsitektur java dapat dilihat pada Gambar 2.1



Gambar 2.5 Arsitektur Java (Martono, 2011)

Arsitektur Java ini merupakan arsitektur J2SDK, yang merupakan salah satu versi Java yang digunakan untuk aplikasi-aplikasi yang berjalan di *Personal Computer* (PC). Lapisan paling bawah pada arsitektur adalah platform, yang merupakan sistem operasi yang menjalankan *Java Virtual Machine* (JVM). Lapisan berikutnya adalah Core APIs dan *Integration APIs* yang merupakan kumpulan *library* yang disediakan Java. Lapisan di atasnya adalah Swing dan AWT yang digunakan untuk membungkus aplikasi agar memiliki tampilan (GUI). Lapisan di atasnya adalah utilitas untuk menjalankan aplikasi Java pada lingkungan *desktop*, dan lapisan paling atas adalah utilitas yang digunakan untuk kompilasi dan *debug*. (Martono, 2011)

2.2.8. System Development Life Cycle

SDLC adalah tahapan-tahapan pekerjaan yang dilakukan oleh analis sistem dan programmer dalam membangun sistem informasi. Langkah yang digunakan meliputi :

1. Melakukan survei dan menilai kelayakan proyek pengembangan sistem informasi
2. Mempelajari dan menganalisis sistem informasi yang sedang berjalan
3. Menentukan permintaan pemakai sistem informasi
4. Memilih solusi atau pemecahan masalah yang paling baik
5. Menentukan perangkat keras (hardware) dan perangkat lunak (software)
6. Merancang sistem informasi baru
7. Membangun sistem informasi baru
8. Mengkomunikasikan dan mengimplementasikan sistem informasi baru

9. Memelihara dan melakukan perbaikan/peningkatan sistem informasi baru bila diperlukan

System Development Lyfe Cycle (SDLC) adalah keseluruhan proses dalam membangun sistem melalui beberapa langkah. Ada beberapa model SDLC. Model yang cukup populer dan banyak digunakan adalah waterfall. Beberapa model lain SDLC misalnya fountain, spiral, rapid, prototyping, incremental, build & fix, dan synchronize & stabilize.

Dengan siklus SDLC, proses membangun sistem dibagi menjadi beberapa langkah dan pada sistem yang besar, masing-masing langkah dikerjakan oleh tim yang berbeda.

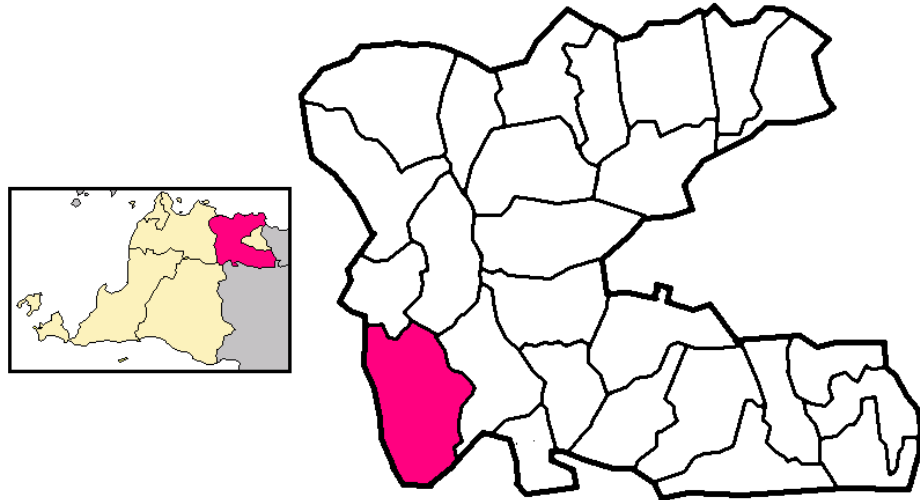
Dalam sebuah siklus SDLC, terdapat enam langkah. Jumlah langkah SDLC pada referensi lain mungkin berbeda, namun secara umum adalah sama. Langkah tersebut adalah

1. Analisis sistem, yaitu membuat analisis aliran kerja manajemen yang sedang berjalan
2. Spesifikasi kebutuhan sistem, yaitu melakukan perincian mengenai apa saja yang dibutuhkan dalam pengembangan sistem dan membuat perencanaan yang berkaitan dengan proyek sistem
3. Perancangan sistem, yaitu membuat desain aliran kerja manajemen dan desain pemrograman yang diperlukan untuk pengembangan sistem informasi
4. Pengembangan sistem, yaitu tahap pengembangan sistem informasi dengan menulis program yang diperlukan

5. Pengujian sistem, yaitu melakukan pengujian terhadap sistem yang telah dibuat
6. Implementasi dan pemeliharaan sistem, yaitu menerapkan dan memelihara sistem yang telah dibuat

Siklus SDLC dijalankan secara berurutan, mulai dari langkah pertama hingga langkah keenam. Setiap langkah yang telah selesai harus dikaji ulang, kadang-kadang bersama expert user, terutama dalam langkah spesifikasi kebutuhan dan perancangan sistem untuk memastikan bahwa langkah telah dikerjakan dengan benar dan sesuai harapan. Jika tidak maka langkah tersebut perlu diulangi lagi atau kembali ke langkah sebelumnya.

2.2.9. Toko Bangunan di Kecamatan Cisoka



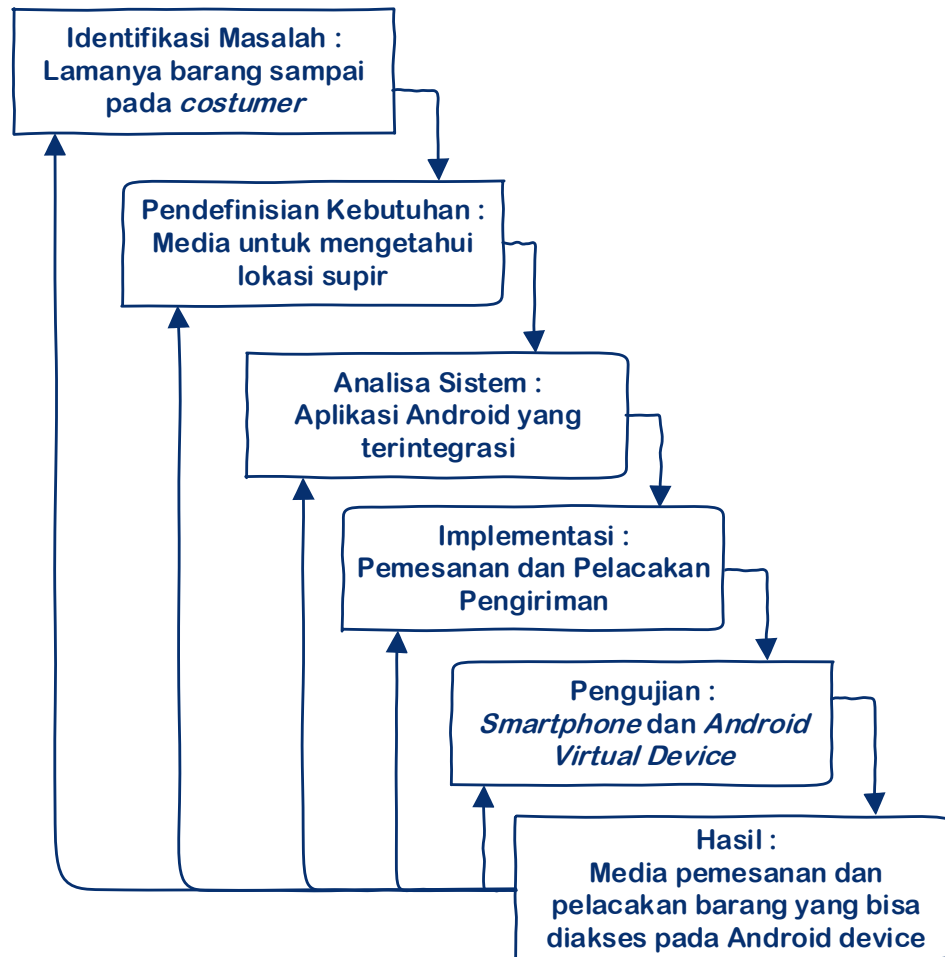
Gambar 2.6 Peta Kecamatan Cisoka

Cisoka adalah sebuah kecamatan di Kabupaten Tangerang, Provinsi Banten, Indonesia. Kecamatan ini terdiri dari 10 desa yaitu: Cisoka, Sukatani, Jeungjing, Selapajang, Caringin, Cibugel, Bojongloa,

Cempaka, Karangharja, dan Carenang. Kecamatan Cisoka merupakan salah satu dari 29 kecamatan yang ada di Kabupaten Tangerang, Provinsi Banten. Luas wilayah Kecamatan Cisoka 26,98 km² dengan jumlah penduduk tahun 2013 sebanyak 86.754, sehingga kepadatan penduduknya 3.215 jiwa per km², hampir mendekati kepadatan penduduk Kabupaten Tangerang 3.291 jiwa per km². Batas wilayah Kecamatan Cisoka meliputi sebelah utara dengan Kecamatan Jayanti dan Balaraja; sebelah timur dengan Kecamatan Tigaraksa dan Solear; sebelah selatan dengan Kecamatan Solear; dan sebelah barat dengan Kabupaten Serang. (Badan Pusat Statistik Kabupaten Tangerang, 2016).

Karena pembangunan yang terjadi di Kecamatan Maja dan Kecamatan Balaraja yang sangat dekat dengan Kecamatan Cisoka maka kecamatan ini perlahan maju. Di website SindoNews.com, Kabupaten Tangerang, Banten, akan memiliki tiga jalan tol. Salah satunya, Tol Serpong-Balaraja. akan melewati sejumlah wilayah kecamatan mulai dari Kecamatan Cisauk, Legok, Panongan, Jambe, Tigaraksa, hingga Balaraja. (Kurniawan, 2017). Sedangkan Maja dalam harian kompas, dipilih para pengembang sebagai wilayah ekspansi bisnis masa depan. Beberapa di antara sekian banyak pengembang yang tertarik dengan Maja adalah Ciputra Group, Hanson Land Group, Kalbe Group, dan Agung Podomoro Group, untuk sekadar menyebut nama. (Alexander, 2014). Bukan karena itu saja, jalur penghubung kedua pembangunan itu pun melewati sepanjang Kecamatan Cisoka.

2.3. Kerangka Pemikiran



Gambar 2.7 Kerangka Pemikiran

Pemesanan dilakukan oleh konsumen dengan menggunakan aplikasi konsumen dan difasilitasi oleh admin dan mengawasinya melalui aplikasi admin. Setiap terjadinya pemesanan, maka konsumen akan diberikan suatu kode pemesanan dan himbauan untuk menyimpan kode tersebut. Admin akan mendapatkan kode order yang berisikan tentang barang yang dipesan, kuantitas barang, nama pelanggan, alamat dan catatan yang di-*input* oleh konsumen. Kode *order* itu akan diberikan kepada supir dan supir akan

menggunakan kode tersebut untuk memberikan koordinat kepada konsumen. Konsumen akan mampu melacak supir beserta kendaraan dan barang pesanan konsumen agar dapat diawasi dengan baik.