

BAB II

LANDASAN TEORI

2.1 Kebakaran

Kebakaran merupakan salah satu prioritas yang dinyatakan oleh Departemen Kehutanan Indonesia dan aksi untuk menangani masalah dimasukkan dalam dokumen komitmen kepada negara-negara donor yang terhimpun dalam *Consultative Group On Indonesia* (CGI). Kebakaran menjadi perhatian Internasional sebagai isu lingkungan dan ekonomi yang dianggap sebagai ancaman potensial bagi pembangunan berkelanjutan karena efeknya secara langsung pada ekosistem dan dampaknya bagi keanekaragaman hayati (UNISDR, 2002).

2.2 Dinas Pemadam Kebakaran Pekanbaru

Badan Penanggulangan Bencana dan Pemadam Kebakaran (BPBPK) Kota Pekanbaru merupakan salah satu lembaga teknis Pemerintah Kota Pekanbaru yang beralamat di Jalan Cempaka no. 31, Kelurahan Pulau Karam, Kecamatan Sukajadi dengan nomor telepon (0761)113-22382 Pekanbaru. Badan Penanggulangan Bencana dan Pemadam Kebakaran (BPBPK) Kota Pekanbaru dibentuk berdasarkan Peraturan Daerah nomor 10 Tahun 2013 Tentang Perubahan Atas Peraturan Daerah Kota Pekanbaru Nomor 9 Tahun 2008 Tentang Pembentukan Susunan Organisasi, Kedudukan Dan Tugas Pokok Lembaga Teknis Daerah Dilingkungan Pemerintah Kota Pekanbaru.

Badan Penanggulangan Bencana dan Pemadam Kebakaran (BPBPK) Kota Pekanbaru mempunyai tugas pokok yaitu melaksanakan sebagian urusan Pemerintah Daerah Kota Pekanbaru di bidang penanggulangan bencana dan pemadam kebakaran. Disamping tugas pokok, Badan Penanggulangan Bencana dan Pemadam Kebakaran (BPBPK) Kota Pekanbaru melaksanakan fungsi Perumusan kebijakan teknis dibidang Pemadam Kebakaran, penyelenggaraan urusan Pemerintah dan Pelayanan Umum, penyusunan rencana kerja, pemantauan dan evaluasi, pembinaan dan pelaporan, penyelenggaraan urusan penatausahaan dinas, dan pelaksanaan tugas-tugas lain.

2.2.1 Visi dan Misi

Visi “ Terciptanya pelayanan kebencanaan dan pemadam kebakaran yang cepat dan profesional bagi masyarakat Kota Pekanbaru”

Misi :

1. Mewujudkan penyelenggaraan pencegahan dan kesiapsiagaan terhadap bencana.
2. Mewujudkan penanganan darurat bencana yang efektif, efisien dan profesional.
3. Mewujudkan pelayanan pemadam kebakaran yang cepat, tepat dan profesional.
4. Mewujudkan sarana prasarana penanggulangan bencana dan pemadam kebakaran yang berkualitas.
5. Mewujudkan pelayanan rehabilitasi dan rekontruksi pembangunan terhadap dampak bencana yang terpadu dan berkelanjutan.
6. Mewujudkan manajemen aparatur yang profesional.

2.2.2 Susunan Organisasi

Badan Penanggulangan Bencana dan Pemadam Kebakaran (BPBPK) Kota Pekanbaru terbagi atas 4 (empat) bidang dan 8 (delapan) seksi yang terdiri dari :

1. Kepala Badan
2. Sekretaris :
 - a. Sub bagian Umum dan Kepegawaian
 - b. Sub bagian Keuangan dan Penatausahaan Aset
 - c. Sub bagian Penyusunan Program
3. Bidang Pencegahan, Kesiapsiagaan, Rehabilitasi dan Rekontruksi :
 - a. Sub bidang Pencegahan, Mitigasi, Pelatihan, Penyuluhan dan kesiapsiagaan
 - b. Sub bidang Rehabilitasi dan Rekontruksi
4. Bidang Kedaruratan dan Logistik :
 - a. Sub bidang Penyelamatan dan Evakuasi
 - b. Sub bidang Distribusi dan Logistik

5. Bidang Pemadam Kebakaran :
 - a. Sub bidang Operasional
 - b. Sub bidang Monitoring dan Evaluasi
6. Bidang Sarana dan prasarana :
 - a. Sub bidang Pemeliharaan Sarana dan Prasarana
 - b. Sub bidang Komunikasi dan Bantuan Teknis
7. Unit Pelaksana Teknis Badan (UPTB)
8. Kelompok Jabatan Fungsional

2.2.3 Sumber Daya Manusia

Sumber daya manusia Badan Penanggulangan Bencana dan Pemadam Kebakaran (BPBPK) Kota Pekanbaru terdiri dari PNS dan Non PNS, dengan rincian sebagai berikut :

Tabel 2.1 Sumber Daya Manusia Berdasarkan Kelompok

No.	Kelompok	Jumlah
1.	PNS Administrasi	17 orang
2.	PNS Operasional	63 orang
3.	THL Operasional (Non PNS)	135 orang
Jumlah		215 orang

(Sumber : Dinas Pemadam Kebakaran Pekanbaru, 2018)

Tabel 2.2 Jumlah PNS Berdasarkan Pendidikan

No.	Tingkat Pendidikan	Jumlah
1.	S2	9 orang
2.	S1	17 orang
3.	D3	2 orang
4.	SMA/SMK/STM	47 orang
5.	SMP/MTs	2 orang
6.	SD	3 orang
Jumlah		80 orang

(Sumber : Dinas Pemadam Kebakaran, 2018)

Tabel 2.3 Jumlah PNS Menurut Golongan

No.	Golongan	Jumlah
1.	IV	11 orang
2.	III	19 orang
3.	II	36 orang
4.	I	4 orang
Jumlah		80 orang

(Sumber : Dinas Pemadam Kebakaran Pekanbaru, 2018)

Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.

b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

Tabel 2.4 Jumlah dan Kualifikasi Pendidikan Tenaga Operasional Kebencanaan dan Kebakaran

No.	Jenis Pendidikan	Jumlah
1.	Pemadam I	125 orang
2.	Pemadam II	2 orang
3.	Resque	38 orang
4.	Inspektur	2 orang
5.	Penyuluh	1 orang
Jumlah		168 orang

(Sumber : Dinas Pemadam Kebakaran Pekanbaru, 2018)

2.2.4 Sarana dan Prasarana Pemadam Kebakaran

Untuk mengatasi bencana kebakaran bangunan gedung dan lahan di Kota Pekanbaru, perlu didukung oleh sarana dan prasarana pemadam kebakaran yang cukup dan layak fungsi, seperti Mobil Pemadam Kebakaran, Pos Pemadam Kebakaran serta peralatan *Safety Personel*. Untuk Pos Pemadam Kebakaran, Pemerintah Kota Pekanbaru memiliki 7 Pos. Sementara itu, Kendaraan Pemadam Kebakaran yang dimiliki Pemerintah Kota Pekanbaru berjumlah 28 Unit yang terdiri dari mobil pemadam kebakaran pompa, rescue, komando dan peralatan, dan sepeda motor. Lokasi pos, jenis kendaraan dan umur kendaraan, sebagaimana tabel berikut:

Tabel 2.5 Kendaraan Operasional Penanggulangan Bencana dan Pemadam Kebakaran yang di Kelompokkan Menurut Jenis

No.	Jenis	Jumlah	Keterangan
1.	MPK/Pompa/Tangki	17 Unit	3000 L = 8 Unit
			4000 L = 4 Unit
			10.000 L = 4 Unit
			12.000 L = 1 Unit
2.	MPK Komando	1 Unit	
3.	Mobil Peralatan	1 Unit	
4.	Mobil Penggerak	1 Unit	
5.	Mobil Ford 4x4 <i>Pick Up Double</i> Kabin	2 Unit	
6.	MPK Tangga	1 Unit	Tangga tidak berfungsi
7.	Mobil <i>Resque</i>	1 Unit	
8.	Sepeda Motor	4 Unit	Mega Pro (Tahun 2005)
			Traiker (Tahun 2014)
Jumlah		28 Unit	

(Sumber : Dinas Pemadam Kebakaran Pekanbaru, 2018)

- Hak Cipta Dilindungi Undang-Undang**
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.
 2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

- Hak Cipta Dilindungi Undang-Undang
1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.
 2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

Tabel 2.6 Pos Pemadam Kebakaran Pekanbaru

No.	Kecamatan	Jumlah
1.	Cempaka	1 Unit
2.	Rumbai Pesisir	1 Unit
3.	Payung Sekaki	1 Unit
4.	Tampan	1 Unit
5.	Bukit Raya	1 Unit
6.	Sail	1 Unit
7.	Tenayan Raya	1 Unit
8.	Sukajadi	1 Unit
Jumlah		8 Unit

(Sumber : Dinas Pemadam Kebakaran Pekanbaru, 2018)

2.2.5 Intensitas Kebakaran

Lajunya pertumbuhan penduduk di Kota Pekanbaru *Equivalent* dengan pertumbuhan perumahan penduduk dan bangunan gedung lainnya, seperti bangunan toko, hotel, restoran, bangunan sekolah dan lain-lain. Salah satu dampak dari pertumbuhan penduduk dan bangunan gedung yang relatif cukup tinggi tersebut adalah meningkatnya kejadian kebakaran di Kota Pekanbaru baik gedung maupun lahan. Kejadian kebakaran bangunan dan lahan tahun 2012 sampai dengan 2017 sebagai mana grafik berikut:



(Sumber : Dinas Pemadam Kebakaran Pekanbaru, 2018)

2.3 Android

Sistem operasi android sistem operasi yang berbasis *Linux Kernel* yang dirancang dan digunakan untuk perangkat bergerak (*taucpad*) seperti telepon pintar (*smartphone*) dan *computer* yang berbasiskan tablet. *Linux Kernel* merupakan layer dimana inti dari sistem operasi android itu berada, yang berisi file-file sistem yang mengatur sistem *processing*, *memory*, *resource*, dan *drivers* (Safaat, 2011). Dengan dukungan finansial dari pihak google, dan kemudian membelinya pada tahun 2005. Antarmuka pengguna android pada umumnya tampilan manipulasi langsung, yang menggunakan gerakan sentuh yang serupa dengan tindakan nyata, seperti contoh menggeser, mencubit, mengetuk yang digunakan untuk memanipulasi *object* di layer android.

Android bertujuan untuk memajukan piranti lunak *mobile* agar setiap pengguna bisa mendapatkan pengalaman baru serta menggali potensi yang ada *paltform* ini (Neven Vrcek, 2009). Pada masa sekarang android merupakan sistem operasi *mobile* yang sangat berkembang. Hal ini merupakan dampak positif dari sifatnya yang *open source* sehingga memberikan kemudahan bagi siapa saja yang ingin mengembangkannya. Ada dua jenis distributor sistem android. Pertama yang mendapatkan dukungan penuh dari Google atau *Google Mall Services* (GMS) dan

kedua adalah yang benar-benar bebas distributornya tanpa dukungan langsung, Google atau dikenal sebagai *Open Handset Distribution* (OHD) (Safaat, 2011).

2.3.1 Activity

Activity adalah digunakan untuk mempresentasikan suatu layer pada *user interface* di sistem operasi android. Bisa juga disebut komponen pada aplikasi android yang menampilkan dan juga untuk mengatur aplikasi sebagai tempat yang digunakan untuk interaksi antara pengguna dengan aplikasi yang ada di android. Contohnya seperti membuat panggilan telepon, mengirim pesan singkat. Sebuah *activity* mengatur satu halaman *user interface* aplikasi. Sehingga jika suatu aplikasi memiliki beberapa *user interface*, berarti bisa dikatakan aplikasi tersebut juga memiliki beberapa *activity*.

2.3.2 Service

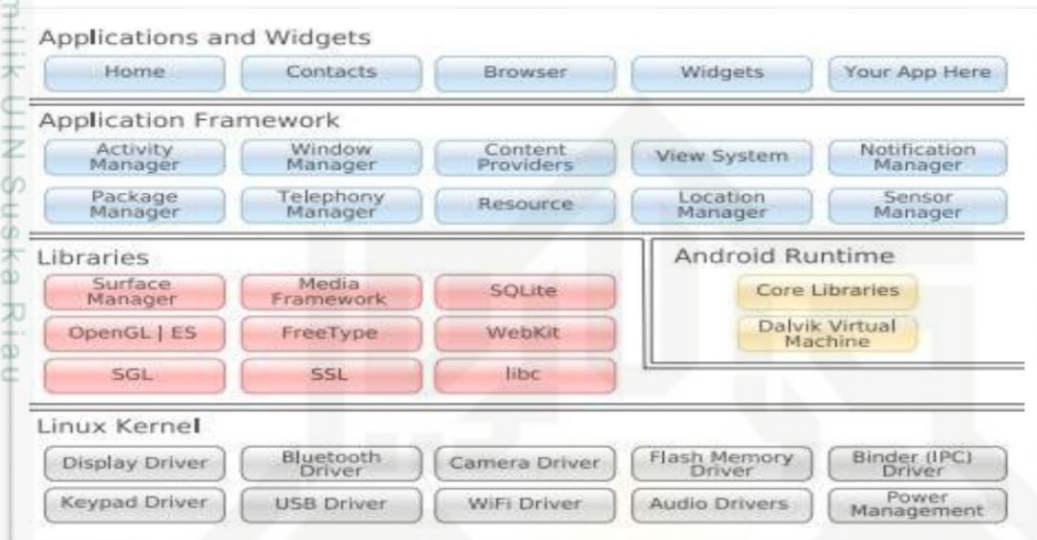
Service adalah komponen aplikasi dalam suatu sistem operasi android yang tidak memiliki UI Visual. Layanan yang digunakan dalam pengolahan bagian dari aplikasi adalah di balik layer. Sementara penggunaan bekerja di layer depan oleh UI.

2.3.3. Intents

Dalam sistem operasi android, sebenarnya *intents* tidak termasuk dalam salah satu komponen aplikasi dari android tersebut, tetapi dalam sistem mekanisme pengerjaannya untuk mengaktifkan suatu komponen kita perlu *intents*. Misalnya jika ingin memanggil suatu aktivitas barulah kita menjalankan sebuah *intents* untuk memanggil aktivitas tersebut.

2.3.4 Arsitektur Android

Secara garis besar, arsitektur android dapat dijelaskan dan ditunjukkan seperti pada Gambar 2.1



Gambar 2.1 Arsitektur Android

(Sumber : Safaat, 2011)

A. *Applications and Widgets*

Applications and widgets ini adalah layer dimana pengguna berhubungan dengan aplikasi saja, biasanya ketika download aplikasi kemudian melakukan instalasi dan jalankan aplikasi tersebut. Pada layer terdapat aplikasi inti termasuk *browser*, klien *email*, program sms (*short message service*), peta, kalender, kontak, dan lain-lain. Semua aplikasi tersebut dibuat dengan menggunakan bahasa pemrograman Java.

B. *Application Framework*

Applications Framework ini adalah layer dimana para pembuat aplikasi dapat melakukan pengembangan aplikasi yang akan dijalankan di sistem operasi Android, karena pada layer inilah aplikasi dapat dirancang dan dibuat, seperti *content-providers* yang berupa sms (*short message service*) dan panggilan telepon.

Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

- a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
- b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

Komponen-komponen yang termasuk di dalam *Applications Frame works* adalah sebagai berikut.

- a. *Views*
- b. *Content Provider*
- c. *Resource Manager*
- d. *Notification Manager*
- e. *Activity Manager*

C. *Libraries*

Libraries ini adalah layer dimana fitur-fitur Android berada, biasanya para pembuat aplikasi mengakses *libraries* untuk menjalankan aplikasinya. Berjalan di atas kernel, layer ini meliputi berbagai *library* C++ ini seperti *Libc* dan *SSL*, serta.

- a. *Libraries* media untuk pemutaran media audio dan video
- b. *Libraries* untuk manajemen tampilan *libraries graphics* mencakup *SGL* dan *OpenGL* untuk grafis 2D dan 3D.
- c. *Libraries SQLite* untuk dukungan database
- d. *Libraries SSL* dan *WebKit* terintegrasi dengan *web browser* dan *security*
- e. *Libraries LiveWebcore* mencakup modern *web browser* dengan *engine embedded web view*.
- f. *Libraries 3D* yang mencakup implementasi *OpenGL ES 1.0 API's*.

D. *Android Run Time*

Layer yang membuat Android dapat dijalankan dimana dalam prosesnya menggunakan implementasi Linux. *Dalvik Virtual Machine* (DVM) merupakan mesin yang membentuk dasar kerangka aplikasi Android. Di dalam *Android Run Time* dibagi menjadi dua bagian yaitu.

- a. *Core Libraries*: aplikasi Android dibangun dalam bahasa Java, sementara Dalvik sebagai virtual mesinnya bukan *Virtual Machine* Java, sehingga diperluas sebuah *libraries* yang berfungsi untuk menerjemahkan bahasa Java/C yang ditangani oleh *Core Libraries*.

Dalvik Virtual Machine: *Virtual* mesin berbasis register yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien dimana merupakan pengembangan

yang mampu membuat Linux kernel untuk melakukan threading dan manajemen tingkat rendah.

2.4 *JavaScript*

JavaScript adalah bahasa yang berbentuk kumpulan skrip berjalan pada suatu dokumen HTML. Bahasa ini adalah bahasa pemrograman untuk memberikan kemampuan tambahan terhadap HTML dengan mengizinkan pengeksekusian perintah-perintah disisi user variabel atau fungsi. *JavaScript* adalah bahasa yang “*case sensitive*” artinya membedakan penamaan variabel dan fungsi yang menggunakan huruf besar dan huruf kecil (Kustiyahningsih, 2011).

2.5 *Web Service*

Web Service adalah sebuah *software* yang dirancang untuk mendukung interoperabilitas interaksi mesin ke mesin melalui sebuah jaringan. *Web Service* secara teknis memiliki mekanisme interaksi antar sistem sebagai penunjang interoperabilitas, baik berupa agregasi (pengumpulan) maupun sindikasi (penyatuan). *Web Service* memiliki layanan terbuka untuk kepentingan integrasi data dan kolaborasi informasi yang bisa diakses melalui internet oleh berbagai pihak menggunakan teknologi yang dimiliki oleh masing-masing pengguna (Sutanta, 2012).

Web Service bertujuan untuk meningkatkan kolaborasi antar pemrograman dan perusahaan, yang memungkinkan sebuah fungsi di dalam *web service* dapat dipinjam oleh aplikasi lain tanpa perlu mengetahui *detail* pemrograman yang terdapat di dalamnya.

2.6 *Global Positioning System (GPS)*

Global Positioning System (GPS) adalah suatu sistem radio navigasi penentuan posisi menggunakan satelit. GPS dapat memberikan posisi suatu objek di muka bumi dengan akurat dan cepat (koordinat tiga dimensi x, y, z) dan memberikan informasi waktu serta kecepatan bergerak secara kontinyu diseluruh dunia (Lengkong, 2015).

GPS (*Global Positioning System*) adalah suatu sistem navigasi menggunakan lebih dari 24 satelit MEO (*Medium Earth Orbit* atau *Middle Earth*

Orbit) yang mengelilingi bumi sehingga penerima-penerima sinyal di permukaan bumi dapat menangkap sinyalnya. GPS mengirimkan sinyal gelombang mikro ke Bumi. Sinyal ini diterima oleh alat penerima di permukaan, dan digunakan untuk menentukan letak, kecepatan, arah, dan waktu. Satelit mengorbit pada ketinggian 12.000 mil di atas bumi dan mampu mengelilingi bumi dua kali dalam 24 jam. Satelit GPS secara kontinyu mengirimkan sinyal radio digital yang mengandung data lokasi satelit dan waktu, pada penerima yang berhubungan. Satelit GPS dilengkapi dengan jam atom yang mempunyai ketepatan waktu satu per satu juta detik. Berdasarkan informasi ini, stasiun penerima mengetahui berapa lama waktu yang digunakan untuk mengirim sinyal sampai kepada penerima di bumi (Abidin, 2007).

Penentuan posisi dengan GPS, pada dasarnya ada dua jenis alat penerima sinyal satelit (*receiver*) GPS yang dapat digunakan, yaitu :

1. Tipe navigasi digunakan untuk penentuan posisi yang tidak menuntut ketelitian tinggi
2. Tipe Geodetik digunakan untuk penentuan posisi yang menuntut ketelitian tinggi

2.7 *Google Maps Application Programming Interface (API)*

API merupakan suatu dokumentasi yang terdiri *interface*, fungsi, kelas, struktur dan sebagainya untuk membangun sebuah perangkat lunak. Bahasa pemrograman yang digunakan oleh *Google Maps* yang terdiri dari HTML, *Javascript* dan AJAX serta XML memungkinkan untuk menampilkan peta *Google Maps* di *website* lain. Google menyediakan layanan *Google Maps API* yang memungkinkan para pengembang untuk mengintegrasikan *Google Maps* ke dalam *website* masing-masing dengan menambahkan data point sendiri (Amri, 2011). *Google Maps API* memberi pengembang banyak kelas fondasi untuk membangun solusi yang kompleks kasus yang berbeda, terutama untuk Sistem Informasi Geografis (SIG) (Davis, 2006).

2.8 *GeoJSON*

GeoJSON adalah format standar terbuka yang dirancang untuk mewakili fitur geografis sederhana, bersama dengan atribut non spasialnya dan format ini

berdasarkan pada format *JavaScript Object Notation* (JSON) (Butler dkk, 2016). Fitur-fitur tersebut meliputi *points* (alamat dan lokasi), *line strings* (jalan dan batas), *polygons* (negara dan provinsi) dan fitur *multipart* dari jenis-jenis ini.

2.9 RAML (*RESTFull API Modelling Language*)

RAML adalah cara untuk mendeskripsikan skema praktikan API *RESTfull* agar lebih mudah untuk dipahami oleh manusia dan komputer. RAML dikatakan sebagai praktikal karena RAML untuk saat ini mengikuti aturan standar pengembangan *RESTfull* API dalam mendeskripsikan *resources*, *methods* *parameters*, *response*, *media types* dan komponen HTTP yang membentuk dasar untuk untuk API pada saat ini. Keuntungan menggunakan RAML adalah pengembang, *partner* dan konsumen API dapat lebih mudah memahami struktur dan spesifikasi dari *RESTfull* API yang tersedia sehingga pengembangan API kedepannya juga akan lebih mudah.

2.10 Pusher

Pusher adalah pemimpin kategori dalam APIs yang menyenangkan bagi pengembang aplikasi yang membangun fitur komunikasi dan kolaborasi.

Implementasi dari *push notification* yang dapat digunakan adalah *Google Cloud Messaging* (GCM). GCM terdiri dari tiga komponen, yaitu aplikasi yang beriperasi pada sistem operasi android yang terhubung dengan GCM, *application server* yang mengirimkan notifikasi dan pesan kepada aplikasi, dan server GCM yang bertugas untuk menyampaikan notifikasi dan pesan dari *application server* (Lee, 2011).

2.11 *Location Based Service* (LBS)

Teknologi *Location Based Service* (LBS) merupakan salah satu bagian dari implementasi mobile GIS yang kecenderungannya memberikan sebuah fungsi terapan sehari-hari yang menampilkan navigasu kendaraan, pencarian alamat suatu daerah (Riyanto, 2010).

Menurut (Fahronzi, 2013) *Location Based Service* (LBS) sebagai layanan informasi dengan memanfaatkan teknologi untuk mengetahui suatu posisi. LBS ini

menggunakan teknologi *Positioning System*, teknologi yang memungkinkan para pengguna dapat memperoleh informasi lokasi sesuai dengan kebutuhannya.

Pada aplikasi *mobile* yang tergantung pada lokasi sebuah *smartphone* yang dapat mengirimkan sebuah informasi tertentu yang di akses dengan perangkat *mobile* melalui jaringan *mobile* tersebut. Dua unsur utama LBS yaitu :

1. *Location Manager* (API Maps) yang menyediakan *tools/resource* untuk LBS, menyediakan fasilitas untuk menampilkan, memanipulasi *maps/peta* beserta *feature-feature* lainnya.
2. *Location Providers* (API Location) yang menyediakan teknologi pencarian lokasi yang digunakan oleh *device/perangkat* dan berhubungan dengan data GPS dan data lokasi real time.

2.11.1 Komponen LBS

Terdapat lima komponen pendukung utama dalam teknologi LBS antara lain:

1. Piranti *Mobile*

Piranti ini berfungsi sebagai alat bantu (*tool*) bagi pengguna untuk meminta informasi. Hasil dari informasi yang diminta dapat berupa teks, suara, gambar dan lain sebagainya. Piranti *mobile* yang dapat digunakan bisa berupa PDA, *smartphone*, laptop. Selain itu, piranti *mobile* dapat juga berfungsi sebagai alat navigasi di kendaraan seperti halnya alat navigasi berbasis GPS.

2. Jaringan komunikasi

Komponen ini berfungsi sebagai jalur penghubung yang dapat mengirimkan data-data yang dikirim oleh pengguna dari piranti *mobile*-nya untuk kemudian dikirimkan ke penyedia layanan dan kemudian hasil permintaan tersebut dikirimkan kembali oleh penyedia layanan kepada pengguna.

3. Komponen *Positioning* (penunjuk posisi/lokasi)

Setiap layanan yang diberikan oleh penyedia layanan biasanya akan berdasarkan pada posisi pengguna yang meminta layanan tersebut. Oleh karena itu diperlukan komponen yang berfungsi sebagai

Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

- Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
- Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

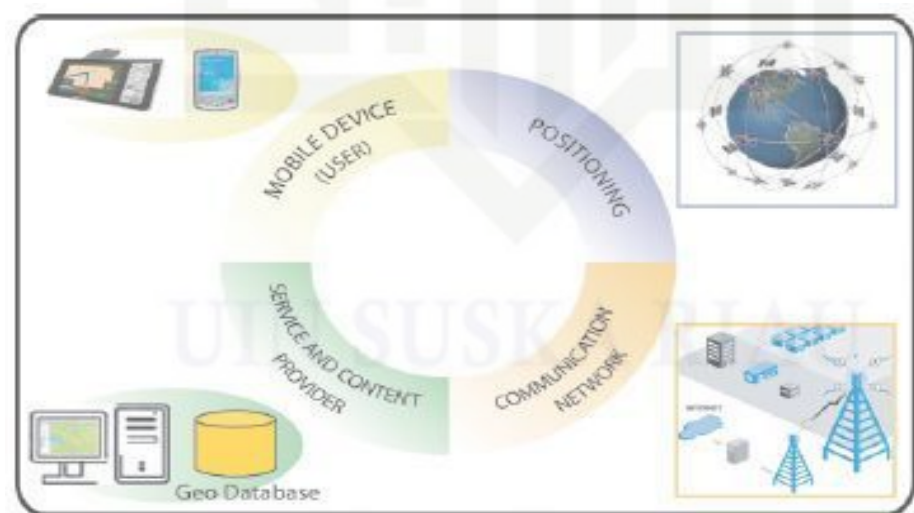
pengolah/pemroses yang akan menentukan posisi pengguna layanan saat itu. Posisi pengguna tersebut bisa didapatkan melalui jaringan komunikasi *mobile* atau juga menggunakan *Global Positioning System* (GPS).

4. Penyedia Layanan dan Aplikasi

Merupakan komponen LBS yang memberikan berbagai macam layanan yang bisa digunakan oleh pengguna. Sebagai contoh ketika pengguna meminta layanan agar bisa tahu posisinya saat itu, maka aplikasi dan penyedia layanan langsung memproses permintaan tersebut, mulai dari menghitung dan menentukan posisi pengguna, menemukan rute jalan dan masih banyak lagi yang lainnya.

5. Penyedia Data dan Konten

Penyedia layanan tidak selalu menyimpan seluruh data dan informasi yang diolahnya. Karena bisa jadi berbagai macam data dan informasi yang diolah tersebut berasal dari pengembang/pihak ketiga yang memang memiliki otoritas untuk menyimpannya. Sebagai contoh basis data geografis dan lokasi bisa saja berasal dari badan-badan milik pemerintah atau juga data-data perusahaan/bisnis/industri maupun perusahaan penyedia data lainnya.



Gambar 2.2 Komponen LBS

(Sumber : Wang, 2008)

2.11.2 Cara Kerja LBS

Cara kerja LBS (*Location Based Service*) yaitu :

1. Fungsi pencarian telah diaktifkan, posisi pengguna sebenarnya dari perangkat *mobile* diperoleh dari *Positioning Service*. Hal ini dapat dilakukan baik oleh perangkat menggunakan GPS sendiri atau layanan posisi jaringan yang berasal dari provider (*Cell Tower*). Setelah itu perangkat *mobile* pengguna mengirimkan permintaan informasi, yang berisi tujuan untuk mencari dan mengirimkan posisi melalui jaringan komunikasi ke *gateway* telekomunikasi.
2. *Gateway* memiliki tugas untuk bertukar pesan di antara jaringan komunikasi selular dan internet dan mengetahui alamat web dari beberapa aplikasi *server* dan *route* permintaan ke spesifik *server* tertentu. *Gateway* akan menyimpan juga informasi tentang perangkat *mobile* yang telah meminta informasi
3. Aplikasi *server* membaca permintaan dan mengaktifkan layanan yang terkait.
4. *Service* menganalisis lagi pesan dan memutuskan mana informasi dan posisi pengguna diperlukan untuk menjawab permintaan pengguna.
5. *Service* akan menemukan bahwa informasi lokasi yang diminta.
6. *Service* akan melakukan *buffer* spasial dan *query routing* untuk mendapatkan beberapa rute terdekat. Setelah itu hasil dikirim kembali ke pengguna melalui internet, *gateway* dan jaringan *mobile*.
7. Kemudian, informasi mengenai lokasi akan disampaikan kepada pengguna baik dalam bentuk peta digital.

2.12 Algoritma

Menurut (Cormen, 2013) Algoritma merupakan prosedur komputasi yang mengambil beberapa nilai atau kumpulan nilai sebagai input kemudian di proses sebagai *output* sehingga algoritma merupakan urutan langkah komputasi yang mengubah input menjadi *output*.

Algoritma memiliki 5 komponen urutan yaitu *finiteness* (terbatas), *definiteness* (kepastian), *input* (masukan), *output* (keluaran), dan *effectiveness*.

Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

- a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
- b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

Dalam merancang algoritma ada tiga komponen yang harus dimiliki yaitu :

1. Komponen masukan (*Input*)
2. Komponen keluaran (*Output*)
3. Komponen Proses (*Processing*)

2.13 Algoritma Dijkstra

Sebuah tulisan Edsger W. Dijkstra pada tahun 1959 seorang ilmuwan komputer mengusulkan algoritma-algoritma untuk solusi dari dua masalah teoritis graf dasar: *the minimum weight*. Algoritma Dijkstra merupakan salah satu cara untuk menemukan masalah jalan terpendek dan merupakan satu dari algoritma-algoritma paling ternama pada ilmu komputer dan sebuah algoritma paling populer pada operasi pencarian adalah pada *link-state routing protocol*, OSPF dan IS-IS.

Algoritma Dijkstra ini juga dapat digunakan untuk menemukan jalur terpendek dari simpul asal ke simpul tujuan dengan cara menghentikan algoritma ketika jalur terpendek ke simpul tujuan telah ditentukan (Rutter, 2009).

Elemen-elemen penyusun prinsip *Greedy* pada Algoritma Dijkstra adalah (Faizah, 2010):

1. Himpunan kandidat. Himpunan ini berisi elemen-elemen yang memiliki peluang untuk membentuk solusi. Pada persoalan lintasan terpendek dalam graf, himpunan kandidat ini adalah himpunan simpul pada graf tersebut.
2. Himpunan solusi. Himpunan ini berisi solusi dari permasalahan yang diselesaikan dan elemennya terdiri dari elemen dalam himpunan kandidat namun tidak semuanya atau dengan kata lain himpunan solusi ini berupa bagian dari himpunan kandidat.
3. Fungsi seleksi. Fungsi seleksi adalah fungsi yang akan memilih setiap kandidat yang memungkinkan untuk menghasilkan solusi optimal pada setiap langkahnya.
4. Fungsi kelayakan. Fungsi kelayakan akan memeriksa apakah suatu kandidat yang telah terpilih (terseleksi) melanggar pembatas atau

tidak. Apabila kandidat melanggar pembatas maka kandidat tidak akan dimasukkan ke dalam himpunan solusi.

5. Fungsi objektif. Fungsi objektif akan memaksimalkan atau meminimalkan nilai solusi. Tujuannya adalah memilih satu saja solusi terbaik dari masing-masing anggota himpunan solusi.

Algoritma ini bertujuan untuk menemukan jalur terpendek berdasarkan bobot terkecil dari satu titik ke titik lainnya. Bobot pada tiap isi dapat berbeda-beda bergantung pada masalah yang dimodelkan dengan *graf*. Bobot dapat menyatakan jarak antara dua buah kota, biaya perjalanan antara dua buah kota, waktu tempuh pesan (*message*) dari sebuah simpul komunikasi ke simpul komunikasi lain (dalam jaringan komputer), ongkos produksi, dan sebagainya (Fitria, 2013).

2.14 Graph

Secara matematis *Graph* G didefinisikan sebagai pasangan himpunan (V, E) , ditulis dengan notasi $G(V, E)$, dimana V adalah sekumpulan dari titik (verteks/*node*) dan E adalah sekumpulan garis (rusuk/*edge*) yang menghubungkan titik yang satu ke titik yang lain. Titik-titik pada *graph* dapat merupakan obyek sembarang seperti kota, atom-atom suatu zat, nama anak, jenis buah, komponen alat elektronik dan sebagainya. *Edge* dapat menunjukkan hubungan sembarang seperti rute penerbangan, jalan raya, sambungan telepon, ikatan kimia, dan lain-lain. Jika terdapat sebuah rusuk e yang menghubungkan titik A dan B , ditulis *edge* (A, B) (Munir, 2005).

Pada suatu *Graph* G terdiri dari dua himpunan yaitu himpunan V dan himpunan E .

- a. *Vertex* (simpul)

V = himpunan simpul yang terbatas dan tidak kosong

- b. *Edge* (sisi/busur)

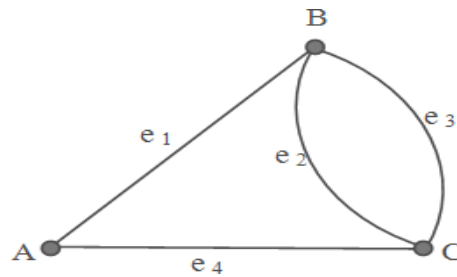
E = himpunan busur yang menghubungkan sepasang simpul.

Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

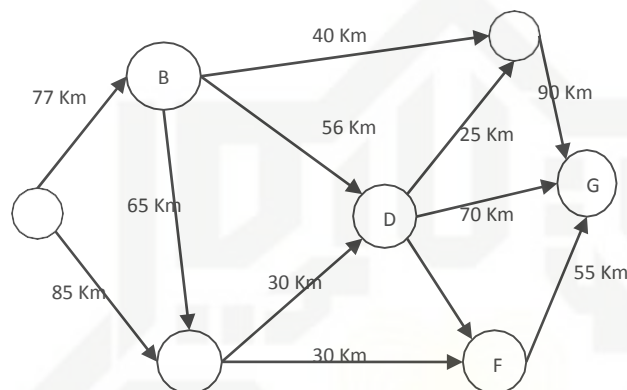
- Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
- Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.



Gambar 2.3 Contoh *Graph*

(Sumber : Mahfudhi, 2010)



Gambar 2.4 Contoh *Graph* Berbobot/berarah

(Sumber : Budiarsyah, 2010)

Pada gambar 2.4 menunjukkan suatu *Graph* ABCDEFG yang berarah dan berbobot dari suatu kota awal A mencari jalur terpendek yang di lewati sampai ke tujuan.

2.15 *Unified Modelling Language (UML)*

UML adalah sebuah alat bantu yang sangat handal di dunia pengembangan sistem berorientasi objek. Hal ini di sebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembangan sistem untuk membuat cetak biru atas visi mereka dalam bentuk baku, mudah di mengerti serta dilengkapi dengan mekanisme efektif untuk berbagi dan mengkomunikasikan rancangan mereka yang lain (Nugroho, 2005).

Tujuan utama perancangan UML adalah:

- Menyediakan bahasa pemodelan Visual yang Ekspresif dan siap pakai untuk mengembangkan dan pertukaran model-model yang berarti.
- Menyediakan mekanisme perluasan dan spesialisasi untuk memperluas konsep inti.

Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.

b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

3. Mendukung spesifikasi independen bahasa pemrograman dan proses pengembangan tertentu.
4. Menyediakan basis formal untuk pemahaman bahasa pemodelan
5. Mendorong pertumbuhan pasar kakas berorientasi objek.
6. Mendukung konsep-konsep pengembangan level lebih tinggi seperti komponen, kolaborasi, *framework* dan *pattern*.

UML menyediakan beberapa diagram yang disediakan dalam UML antara lain:

- a. Diagram *use case* (*Use case diagram*)
- b. Diagram aktivitas (*activity diagram*)
- c. Diagram Sekuensial (*sequence diagram*)
- d. Diagram kolaborasi (*collaboration diagram*)
- e. Diagram kelas (*class diagram*)
- f. Diagram *statechart* (*statechart diagram*)
- g. Diagram komponen (*component diagram*)
- h. Diagram *deployment* (*deployment diagram*)

2.15.1 Diagram-Diagram UML

UML memiliki beberapa diagram yang digunakan untuk menggambarkan suatu sistem. Tujuan pembuatan diagram ini adalah agar sistem mudah dimengerti oleh semua pihak, baik yang teknis maupun nonteknis (Fowler, 2005). Beberapa contoh dari diagram tersebut, antara lain:

1. *Class Diagram*

Diagram ini terdiri dari sekumpulan class dan interface lengkap dengan kolaborasi dan hubungan antara mereka.

2. *Use Case Diagram*

Menggambarkan sekumpulan *use case* dan *actor* dan hubungan antara mereka. *Use case* diagram mempunyai peranan penting dalam pengorganisasian dan pemodelan behavior dari sistem.

3. Activity Diagram

Menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang.





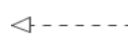
4. Sequence Diagram

Merupakan diagram interaksi yang menekankan pada urutan waktu dari pertukaran message.



a. Diagram Kelas (*Class Diagram*)

Diagram kelas digunakan untuk menampilkan kelas-kelas atau paket-paket di dalam sistem dan relasi antar mereka. Ia memberikan gambaran sistem secara statis. Biasanya, dibuat beberapa diagram kelas untuk satu sistem.

Tabel 2.7 Simbol Diagram Kelas

No	Gambar	Nama	Keterangan
1		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>).
2		<i>Nary Association</i>	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		<i>Class</i>	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		<i>Collaboration</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
5		<i>Realization</i>	Operasi yang benar-benar dilakukan oleh suatu objek.

Tabel 2.7 Simbol Diagram Kelas (Lanjutan)

6		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan memengaruhi elemen.
7		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya





b. Diagram *Use Case* (*Use Case Diagram*)

Diagram *use case* menyajikan interaksi antara *use case* dan aktor. Dimana, aktor dapat berupa orang, peralatan, atau sistem lain.

Tabel 2.8 Simbol Diagram *Use Case*

No	Gambar	Nama	Keterangan
1		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan <i>use case</i> .
2		<i>Dependency</i>	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (<i>independent</i>).
3		<i>Generalization</i>	Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (<i>ancestor</i>).
4		<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara <i>eksplisit</i> .
5		<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
6		<i>Association</i>	Apa yang menghubungkan antara objek satu dengan objek lainnya.




Tabel 2.8 Simbol Diagram *Use Case* (Lanjutan)

7		<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		<i>Use Case</i>	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
9		<i>Collaboration</i>	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).
10		<i>Note</i>	Elemen fisik yang eksis saat aplikasi dijalankan dan mencerminkan suatu sumber daya komputasi



c. Diagram Aktivitas (*Activity Diagram*)

Diagram aktivitas atau *activity* diagram menggambarkan aliran fungsionalitas sistem. Pada tahap pemodelan bisnis, diagrama aktivitas dapat digunakan untuk menunjukkan aliran kerja bisnis (*business work flow*). Dapat juga digunakan untuk menggambarkan aliran kejadian (flow of event) dalam *use case*.

Tabel 2.9 Simbol Diagram Aktifitas

No	Gambar	Nama	Keterangan
1		<i>Activity</i>	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		<i>Action</i>	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		<i>Initial Node</i>	Bagaimana objek dibentuk atau diawali.

Tabel 2.9 Simbol Diagram Aktifitas (Lanjutan)

4		<i>Activity Final Node</i>	Bagaimana objek dibentuk dan dihancurkan
5		<i>Fork Node</i>	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran

d. Diagram Sekuensial (*Sequence Diagram*)

Diagram sekuensial atau *sequence diagram* digunakan untuk menunjukkan aliran fungsionalitas dalam *use case*.

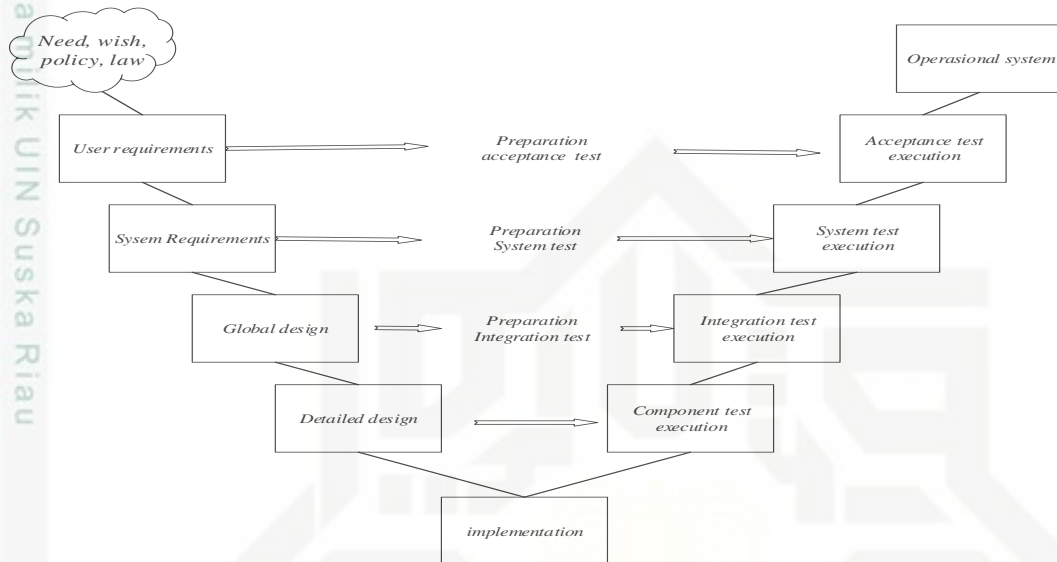
Tabel 2.10 Simbol Diagram Sekuensial

No	Gambar	Nama	Keterangan
1		<i>Object</i>	<i>Object</i> merupakan <i>instance</i> dari sebuah class dan dituliskan tersusun secara horizontal. Digambarkan sebagai sebuah <i>class</i> (kotak) dengan nama obyek didalamnya yang diawali dengan sebuah titik koma
2		<i>Actor</i>	<i>Actor</i> dapat berkomunikasi atau berinteraksi dengan sistem.
3		<i>Lifeline</i>	<i>Lifeline</i> mengindikasikan keberadaan sebuah <i>object</i> dalam basis waktu. Notasi untuk <i>Lifeline</i> adalah garis putus-putus <i>vertical</i> yang ditarik dari sebuah obyek.
4		<i>Activation</i>	<i>Activation</i> dinotasikan sebagai sebuah kotak segi empat yang digambar pada sebuah <i>lifeline</i> . <i>Activation</i> mengindikasikan sebuah obyek yang akan melakukan sebuah aksi.
5		<i>Message</i>	<i>Message</i> , digambarkan dengan anak panah horizontal antara <i>Activation</i> . <i>Message</i> mengindikasikan komunikasi antara <i>object-object</i> .

2.16 Metode Pengembangan Sistem V-Model (Graham et.al, 2006)

Metode pengembangan sistem *V Model* merupakan perluasan dari metode *waterfall*. Disebut sebagai perluasan karena tahapan-tahapannya mirip dengan

yang terdapat dalam metode *waterfall*. Jika dalam metode *waterfall* proses dijalankan secara linear maka dalam *V Model* proses dilakukan bercabang. Dalam *V Model* ini menggambarkan hubungan antara tahap pengembangan *software* dengan tahap pengujiannya.



Gambar 2.5 *V Model Life Cycle*

(Sumber: Graham et.al, 2006)

Proses Pengembangan ini seimbang dan bergantung pada verifikasi dari langkah sebelumnya sebelum dilanjutkan. Hasil dari setiap tahap perlu diperiksa dan disetujui sebelum dilanjutkan (Balaji, 2012).

Berikut penjelasan masing-masing tahap beserta tahap pengujiannya.

1. *User Requirement & Acceptance Testing*

Tahap *User Requirement* sama seperti yang terdapat dalam model *waterfall*. Hasil dari tahap ini adalah dokumentasi kebutuhan pengguna.

Rumus mencari presentase :
$$\frac{\text{jumlah jawaban persoal} \times \text{bobot}}{\text{Point tertinggi} \times \text{jumlah responden}} \times 100\%$$

Rumus mencari presentase keseluruhan :
$$\frac{\text{total presentase}}{100\% \times \text{jumlah pertanyaan}} \times 100\%$$

Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:
 - a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
 - b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.
2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

2. *System Requirements & System Testing*

Dalam tahap ini analisis sistem mulai merancang sistem dengan mengacu pada dokumentasi kebutuhan pengguna yang sudah dibuat pada tahap sebelumnya. Keluaran dari tahap ini adalah spesifikasi *software* yang meliputi organisasi sistem secara umum, struktur data, dan yang lain.

3. *Global Design & Integration Testing*

Sering juga disebut *Hight Level Design*. Dasar dari pemilihan arsitektur yang akan digunakan berdasarkan kepada beberapa hal seperti pemakaian kembali tiap modul, ketergantungan tabel dalam basis data, hubungan antar *interface*, detail teknologi yang dipakai. Penyajian pada tahap desainpun berbagai macam, seperti menggunakan UML dan diagram.

4. *Detail Design & Unit Testing*

Perancangan dipecah menjadi modul-modul yang lebih kecil. Setiap modul tersebut diberi penjelasan yang cukup untuk memudahkan *programmer* melakukan *coding*. Tahap ini menghasilkan spesifikasi program seperti fungsi dan logika tiap modul, pesan kesalahan, proses *input* dan *output* untuk tiap modul, dan lain-lain.

5. *Implementation*

Design yang telah dirancang kemudian ditranslasikan kedalam kode melalui *event-event* untuk mengimplementasikan logika program. Proses implementasi ini dilakukan pada perangkat lunak pengembangan.

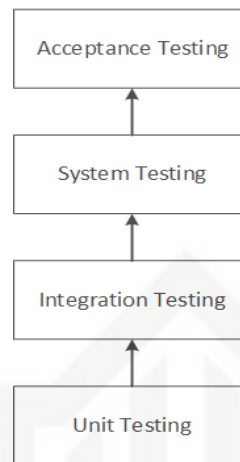
Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

- Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
- Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

Menurut (Rifai, 2015) terdapat empat fase pengujian yang dapat dilihat pada Gambar 2.6.



Gambar 2.6 Tahapan Fase Pengujian
(Sumber: Rifai, 2015)

1. Unit Testing

Unit Testing adalah proses pengujian perangkat lunak dimana masing-masing unit atau komponen diuji. Tujuannya adalah untuk memvalidasi bahwa setiap unit perangkat lunak sudah melakukan seperti apa yang telah dirancang.

Menurut (Pressman, 2010) *unit testing* berfokus pada upaya verifikasi terhadap unit terkecil dari perancangan perangkat lunak. Pengujian unit berfokus pada logika pemrosesan internal dan struktur data didalam komponen.

Unit testing merupakan proses dimana pengujian dilakukan pada bagian *basic* dari kode program. Contohnya adalah memeriksa kode program pada *event*, *procedure*, dan *function*. *Unit testing* meyakinkan bahwa masing-masing unit tersebut berjalan sebagaimana mestinya.

2. Integration Testing

Integration Testing adalah proses pengujian perangkat lunak dimana unit individu digabungkan dan diuji sebagai sebuah kelompok. Sehingga pengujian ini mampu menampilkan kesalahan dalam interaksi antar unit. Menurut (Pressman, 2010) pengujian integrasi adalah teknik untuk membangun arsitektur perangkat lunak, sementara pada saat yang sama

Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

- a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.
- b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.

melakukan pengujian untuk menemukan kesalahan terkait antarmuka. Tujuannya adalah untuk mengambil komponen yang diuji dan membangun struktur program yang telah ditentukan oleh perancangan.

Setelah melakukan *Unit* dan *Component Testing*, langkah berikutnya adalah memeriksa bagaimana unit-unit tersebut bekerja sebagai suatu kombinasi, bukan lagi sebagai suatu unit yang individual. Sebagai contoh, kita memiliki sebuah proses yang dikerjakan oleh dua *function*, di mana satu *function* menggunakan hasil output dari *function* yang lainnya.

Pada tahap *Integration Testing*, kita memeriksa hasil dari interaksi kedua *function* tersebut, apakah bekerja sesuai dengan hasil yang diharapkan. Kita juga harus memastikan bahwa seluruh kondisi yang mungkin terjadi dari hasil interaksi antar unit tersebut menghasilkan *output* yang diharapkan.

3. *System Testing*

System Testing adalah proses pengujian dimana perangkat lunak yang diuji sudah lengkap dan terintegrasi. Tujuan dari pengujian ini adalah untuk mengevaluasi kesesuaian sistem dengan persyaratan yang telah ditentukan.

4. *Acceptance Testing*

Acceptance Testing atau uji penerimaan adalah pengujian formal dilakukan untuk menentukan apakah sistem menerima kriteria penerimaan dan memastikan jika pengguna dapat menerima sistem. Tujuan dari pengujian ini adalah untuk mengetahui tingkat kelayakan dari perangkat lunak.

Seperti *Integration Testing*, *Acceptance Testing* juga meliputi pengujian keseluruhan aplikasi. Perbedaannya terletak pada siapa yang melakukan testing. Pada tahap ini, *end user* yang terpilih melakukan testing terhadap fungsi-fungsi aplikasi dan melaporkan permasalahan yang ditemukan. Proses ini merupakan salah satu tahap final sebelum pengguna menyetujui dan menerima penerapan sistem aplikasi yang baru.

2.17 Metode Pengujian *Blackbox Testing*

Teknik pengujian *blackbox* berfokus pada domain informasi dari perangkat lunak, dengan melakukan *test case* dengan mempartisi domain *input* dari suatu program dengan cara yang memberikan cakupan pengujian yang mendalam. *Blackbox* merupakan metode pengujian perangkat lunak yang dilakukan hanya mengamati hasil eksekusi data uji dan memeriksa fungsional dari perangkat lunak. Cakupan pengujian yang dilakukan pada *blackbox testing* adalah perihal pengujian *interface* dan *form validation*. Pengujian *blackbox testing*, dilakukan untuk menemukan hal-hal sebagai berikut:

1. Fungsi yang tidak benar atau tidak ada.
2. Kesalahan antarmuka (*Interface errors*).
3. Kesalahan pada struktur data dan akses *database*.
4. Kesalahan performansi (*Performance errors*),
5. Kesalahan inisialisasi dan terminasi.

2.18 Penelitian Terdahulu

Penelitian terdahulu ini menjadi salah satu acuan penulis dalam melakukan penelitian sehingga penulis dapat memperbanyak referensi yang digunakan dalam mengkaji penelitian yang dilakukan. Untuk melihat perbandingan penelitian sebelumnya maka dapat dilihat pada tabel 2.11 di bawah ini:

Tabel 2.11 Penelitian Terdahulu

No	Nama Peneliti	Judul Penelitian	Hasil Penelitian
1.	Cahyani Budihartanti, Riswan Pandiangan (2016)	Rancang bangun aplikasi android pencarian rumah sakit di Jakarta menggunakan Algoritma Dijkstra	Menghasilkan aplikasi dapat menampilkan rute jalan ke Rumah Sakit dari posisi user serta jaraknya sekarang ini dalam bentuk peta.
2.	Muhammad Khoiruddin Harahap, Nurul Khairina (2017)	Pencarian jalur terpendek dengan Algoritma Dijkstra	Menghasilkan aplikasi jalur terpendek ditentukan dari sebuah jalur perjalanan dengan menentukan <i>vertex</i> awal dan <i>vertex</i> tujuan.

Tabel 2.11 Penelitian Terdahulu (Lanjutan)

Hak Cipta Dilindungi Undang-Undang 1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber: a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah. b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau. 2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.	3.	Windi Eka Yulia R., Dwiretno Istiadi, Abdul Roqib (2015)	Pencarian SPBU terdekat dan penentuan jarak terpendek menggunakan Algoritma Dijkstra	Menghasilkan aplikasi perhitungan jarak terpendek dalam pencarian SPBU dipengaruhi oleh nilai kriteria, <i>cost</i> , dan <i>reserse_cost</i> , nilai waktu tempuh di dapat dari perhitungan jarak dibagi dengan kecepatan, dan dari 33 SPBU di dapat 19 menjual pertamax, 7 menjual pertamax dex, 6 menjual keduanya.
	4.	Muhammad Syamsuddin Yusuf, Hanifah Muslimah Az-Zahra, Diah Harnoni Apriyanti (2017)	Implementasi Algoritma Dijkstra dalam menemukan jarak terdekat dari lokasi pengguna ke tanaman yang dituju berbasis android	Menghasilkan implementasi Algoritma Dijkstra dalam menentukan jarak terpendek dari lokasi pengguna ke tanaman yang dituju menghasilkan nilai <i>usability</i> sebesar 70,916% dengan rentang interval 60% - 80% yang dapat di kategorikan sebagai aplikasi yang memuaskan.
	5.	Siti Nandiroh, Haryanto, Hafidh Munawir (2013)	Implementasi Algoritma Dijkstra sebagai solusi efektif pembuatan sistem bantuan bencana <i>real time</i>	Menghasilkan terciptanya sistem layanan informasi yang real time, dan sistem navigasi yang bisa diakses dengan telpon seluler yang mampu menunjukkan <i>route</i> yang paling pendek, serta dapat menunjukkan <i>route</i> alternatif jika terjadi kemacetan atau terjadi kecelakaan di salah satu ruas jalan atau di suatu lokasi bencana.
	6.	Luthfi Fahronzi (2013)	<i>Aplikasi Location Based Service</i> (LBS) untuk pencarian rute terpendek menggunakan Algoritma Dijkstra	Menghasilkan aplikasi yang berhasil koneksi ke <i>server</i> , menampilkan fitur dan konten yang terdapat pada <i>server</i> , menampilkan <i>route</i> jalan ke lokasi <i>customer</i> , memberikan informasi jarak tempuh dan waktu tempuh, dan menampilkan urutan daftar <i>customer</i> berdasarkan perhitungan Dijkstra.

Tabel 2.11 Penelitian Terdahulu (Lanjutan)

7.	Shaga Bogas Priatmoko (2014)	Algoritma Dijkstra untuk pencarian jalur terdekat dan rekomendasi objek pariwisata di pulau Bali	Menghasilkan aplikasi yang digunakan sebagai alat untuk memudahkan penentuan jadwal perjalanan pariwisata, menentukan keputusan pemilihan lokasi pariwisata.
8.	Fitria, Apri Triansyah (2013)	Implementasi Algoritma Dijkstra dalam aplikasi untuk menentukan lintasan terpendek jalan darat antar kota di Sumatera Bagian Selatan	Menghasilkan sebuah aplikasi yang digunakan untuk mencari rute terpendek secara optimal, memudahkan dalam penyusunan peta secara dinamik apabila terdapat perubahan kondisi peta, program dapat menyesuaikan dengan kondisi peta yang terbaru.

Hak Cipta Dilindungi Undang-Undang

1. Dilarang mengutip sebagian atau seluruh karya tulis ini tanpa mencantumkan dan menyebutkan sumber:

a. Pengutipan hanya untuk kepentingan pendidikan, penelitian, penulisan karya ilmiah, penyusunan laporan, penulisan kritik atau tinjauan suatu masalah.

b. Pengutipan tidak merugikan kepentingan yang wajar UIN Suska Riau.

2. Dilarang mengumumkan dan memperbanyak sebagian atau seluruh karya tulis ini dalam bentuk apapun tanpa izin UIN Suska Riau.