BAB 2
TINJAUAN PUSTAKA

2.1 Profil Tempat Penelitian

Profil tempat penelitian berisi tentang informasi tempat penelitian yang dalam hal ini adalah PD. Kebersihan Kota Bandung. Berawal dari sejarah berdirinya perusahaan hingga tujuan-tujuan yang hendak dicapai oleh perusahaan. Profil peruahaan juga menjabarkan struktur organisasi serta sistem kerja perusahaan.

2.1.1 Sejarah PD. Kebersihan Kota Bandung

Perusahaan Daerah (PD) Kebersihan Kota Bandung merupakan Badan Usaha Milik Daerah (BUMD) yang bergerak di bidang jasa kebersihan yang didirikan 1985 silam oleh Wali Kota Bandung saat itu, Bapak Ateng Wahyudi. Pendirian PD Kebersihan mengacu pada Peraturan Daerah (Perda) Kotamadya Daerah Tingkat II Bandung Nomor 02/PD/1985 sebagaimana telah diubah terakhir dengan Perda Kota Bandung Nomor 14/2011 tentang Perusahaan Daerah Kebersihan. Adapun tugas pokok PD Kebersihan adalah menyelenggarakan pelayanan jasa di bidang persampahan untuk mewujudkan kota yang bersih, yang kegiatannya meliputi penyapuan jalan, pengumpulan dan pemindahan, pengolahan, pengangkutan, dan pemrosesan akhir sampah. Berdasarkan Perda Kota Bandung Nomor 14/2011, tujuan pendirian PD Kebersihan adalah menyelenggarakan usaha berupa penyediaan pelayanan jasa pengolahan sampah, pengelolaan dan pemanfaatan sampah, pelayanan kebersihan, perbengkelan sarana pengelolaan sampah, dan usaha lainnya yang ditetapkan dengan keputusan direksi. Selain itu, PD Kebersihan juga melaksanakan penugasan pemerintah daerah di bidang pengelolaan sampah dalam rangka memberikan pelayanan kebersihan kepada masyarakat dan memberikan kontribusi kepada pendapatan asli daerah (PAD).

PD Kebersihan Kota Bandung didirikan melalui fase-fase yang dimulai sejak 1960 hingga saat ini. Fase-fase tersebut secara garis besar terbagi ke dalam

lima periode. Pada periode 1960 sampai dengan 1967, pengelolaan dan penanganan kebersihan sudah menjadi perhatian pemerintah daerah, yang pada masa itu menjadi tanggung jawab Tim Pembersihan dan Pertamanan Kota (TPPK) yang menginduk pada Unit Kerja Dinas Teknik A. Pada periode 1967 sampai dengan 1972, beban pengelolaan dan penanganan kebersihan serta pertamanan kota bertambah seiring bergabungnya Bagian *Riool* dan Saluran Terbuka serta Dinas Pekerjaan Umum dan Dinas Teknik A. Pada periode 1972 sampai dengan 1983, pemerintah daerah memandang perlu mengembangkan institusi dengan memisahkan penanganan kebersihan, pertamanan, *rioolering*, dan saluran terbuka dari Dinas Teknik Penyehatan seiring dengan bertambahnya volume pekerjaan di bidang kebersihan dan meningkatnya kebutuhan air minum serta semakin pesatnya pertumbuhan dan perkembangan kota. Dengan pemikiran tersebut, maka pada 1972, dibentuklah unit kerja baru, yakni Dinas Kebersihan dan Keindahan Kota (DK3) Kotamadya Daerah Tingkat II Bandung. Dengan terbentuknya DK3, penanganan dan pengelolaan kebersihan mulai ditangani unit kerja tersendiri, meskipun di dalamnya masih harus menangani pertamanan, *riool*, dan saluran. Pada periode 1983 sampai dengan 1985, bobot pekerjaan masing-masing bagian terus meningkat, sehingga volume pekerjaan DK3 bertambah padat dan kompleks. Kondisi itu sejalan dengan tuntutan warga Kota Bandung yang terus meningkat, baik dalam pelayanan kebersihan maupun terpeliharanya sungai dan saluran.

Diperlukan sistem modern, meskipun diperlukan dana yang tidak sedikit, seperti untuk pengadaan saranadan prasarananya. Untuk kebutuhan tersebut, masyarakat dinilai perlu diberdayakan agar berperan aktif, baik dalam dukungan dana maupun penanganan kebersihan. Dengan demikian, penanganan kebersihan secara profesional diharapkan betul-betul tercapai dankebersihan kota pun terpelihara baik. Atas pertimbangan tersebut, dibentuklah PD Kebersihan pada tahun 1985 sebagai perusahaan daerah pertama yang sekaligus dijadikan pilot project di Indonesia dalam hal penanganan dan pengelolaan kebersihan oleh pemerintah daerah. Alasan lain yang melatarbelakangi pembentukan PD Kebersihan, antara lain meningkatkan kualitas pelayanan dalam bidang kebersihan dengan tersedianya prasarana, sarana, dan peralatan yang lebih modern; upaya

membuka lapangan kerja bagi warga Kota Bandung; menggali sumber pendapatan daerah dengan cara memberdayakan masyarakat untuk berpartisipasi aktif menangani permasalahan kebersihan, melalui dukungan dana lewat pembayaran jasa pelayanan kebersihan. Selain itu, pembentukan PD Kebersihan juga menjadi upaya yang harus ditempuh pemerintah daerah untuk mengurangi beban anggaran karena penanganan kebersihan memerlukan dana yang sangat besar. Dengan dikelola oleh perusahaan daerah, operasional penanganan dan pengelolaan kebersihan diharapkan dapat dibiayai secara mandiri dan secara bertahap diharapkan memberikan kontribusi terhadap PAD dari sebagian laba yang diperolehnya.

2.1.2 Logo Instansi

Logo adalah lambang dari sebuah perusahaan yang juga merupakan simbol yang memberi penjelasan tentang citra identitas dari perusahaan. Dengan memiliki logo maka perusahaan menempatkan dirinya secara berbeda dalam masyarakat dan konsumen.

Gambar 2.1 Logo Instansi

Image Kebersihan Kota Bandung diibaratkan dengan landscape Kota Bandung yang terdiri dari gunung, air, dan udara. Di dalamnya terdapat “Gunung (Tangkuban Perahu)”, “Sungai yang Bersih”, dan “Udara Sejuk”. Kemudian sebagai suatu kesatuan logo, terbentuklah burung cangkurileung yang bermakna damai. Hal ini merupakan harapan pula bagi kita akan terciptanya kedamaian setelah terwujud lingkungan yang nyaman.

Warna yang digunakan pada logo Bandung Resik ini didominasi oleh warna biru muda dan hijau muda. Warna biru melambangkan gunung, udara yang sejuk, sedangkan warna hijau melambangkan keindahan, kebersihan, ke asrian dan kenyamanan, Sehingga diharapkan menciptakan kedamaian setelah melihatnya dan terwujudnya lingkungan yang nyaman. Warna kuning pada biji padi melambangkan kesejahteraan, lambang ini melambangkan simbol pada PD Kebersihan yang memberikan Pendapatan Asli Daerah ke Pemerintah Kota Bandung.

2.1.3 Visi dan Misi

Sebuah perusahaan yang besar pastilah memiliki visi jangka panjang serta misi utama yang kuat. Visi dan Misi akan menentukan arah gerak dan peta jalan sebuah perusahaan. Visi adalah tujuan besar dari berdirinya sebuah perusahaan, sedangkan misi adalah upaya-upaya yang dilakukan untuk mewujudkan tujuan besar tersebut.

2.1.3.1 Visi

Visi PD kebersihan adalah menjadi perusahaan **profesional** dan memberikan **solusi inovatif** dalam pelayanan kebersihan kepada masyarakat.

a. **Profesional** di bidang pelayanan kebersihan kota : sebagai perusahaan daerah PD Kebersihan berkomitmen mempunyai kompetensi yang unggul untuk menghasilkan kerja yang tuntas dan berkualitas.
b. **Solusi inovatif** di bidang pelayanan kebersihan kota: sebagai perusahaan daerah PD Kebersihan senantiasa mengembangkan kreatifitas dan inovasi serta bertumbuh sehingga kehadirannya bisa memberi solusi yang bisa diterapkan oleh kota Bandung maupun kota-kota lain di Indonesia.

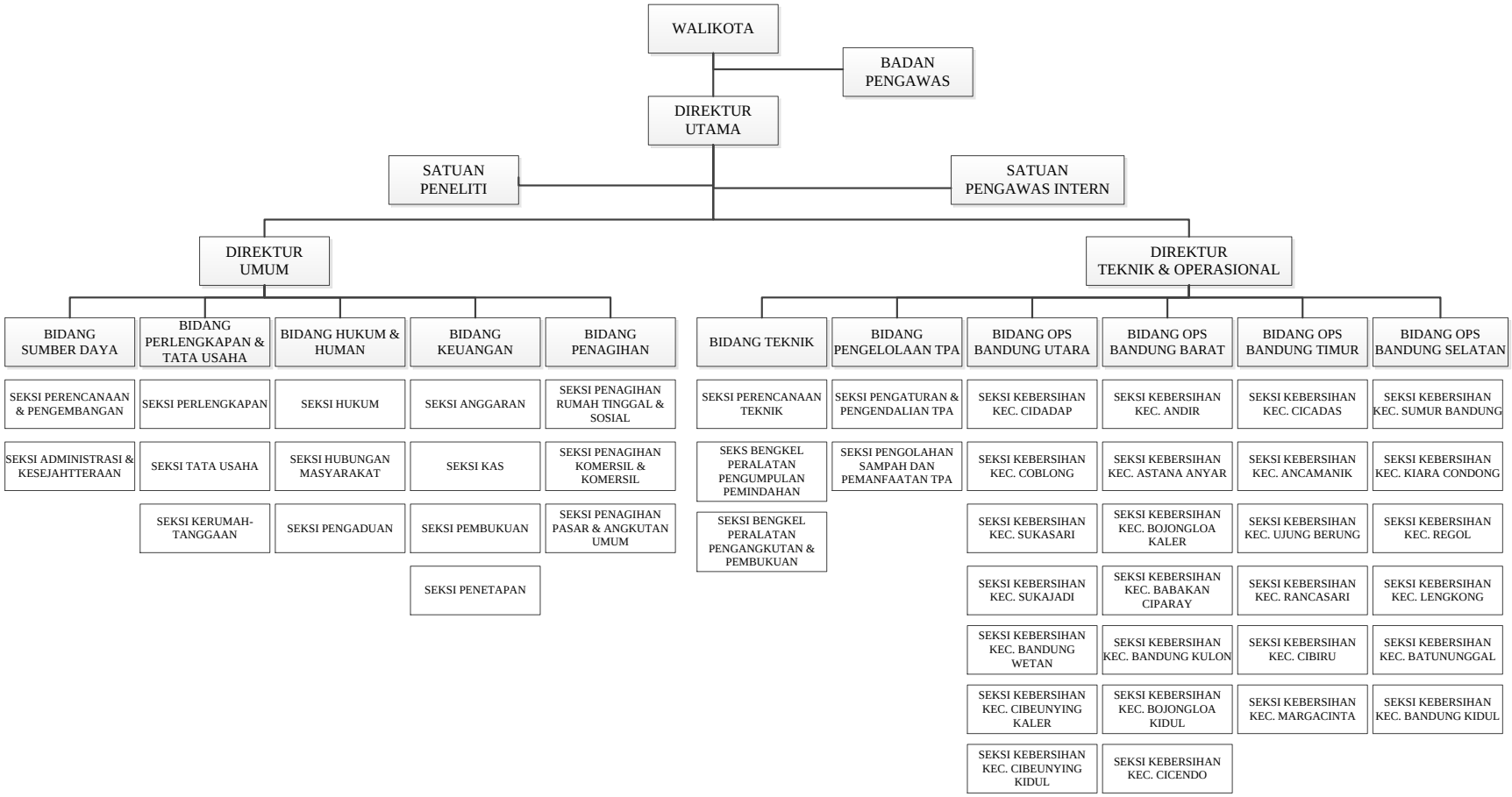
2.1.3.2 Misi

Misi PD Kebersihan adalah **CSR (*Collaboration, Service Excellent, Role Model***):

a. ***Collaboration***, yang berarti Mengembangkan kemitraan dan kolaborasi yang saling menguntungkan dan bermanfaat baik dalam kegiatan bisnis maupun kegiatan pengelolaan lingkungan bagi masyarakat secara luas
b. ***Service Excellent***, yang berarti Memberikan pelayanan yang unggul dalam pelayanan kebersihan kota untuk memuaskan kepada semua stakeholder& masyarakat Kota Bandung serta
c. ***Role Model*** (*Best Practice*), yang berarti Menjadi percontohan pelayanan kebersihan kota di Indonesia.

Struktur Organisasi

Untuk menjalankan roda perusahaan, berdasarkan Peraturan Walikota nomor 101 tahun 2006 tentang Susunan Organisasi dan Tata Kerja (SOTK) Perusahaan Daerah Kebersihan Kota Bandung, Struktur Organisasi di PD Kebersihan Kota Bandung adalah sebagai berikut .



Gambar 2.2 Struktur Organisasi PD Kebersihan Kota Bandung

Dalam struktur pemerintahan Kota Bandung, PD Kebersihan berada langsung di bawah Walikota Bandung. Sekertaris Daerah & Bagian Ekonomi menjadi pembina sekaligus jalur koordinasi dalam pelaksanaannya. Sebagai pemilik, Walikota Bandung menunjuk 3 orang sebagai Badan Pengawas.

Struktur organisasi inti terdiri dari 3 direksi (Direktur Utama, Direktur Umum dan Direktur Teknik) & 13 Bidang.

a. Direktur utama (Dirut) membawahi 2 bidang yaitu Bidang Satuan Penelitian dan Satuan Pengawas Internal.
b. Direktur Umum (Dirum) membawahi 5 bidang yaitu Bidang SDM, Bidang Penagihan, Bidang Hukum Humas, Bidang Perlengkapan Tata Usaha dan Bidang Keuangan.
c. Direktur Teknik Operasional (DTO) membawahi 6 bidang yaitu Bidang Teknik, Bidang Tempat Pemrosesan Akhir (TPA).

Di tahun 2016 Direksi membuat tambahan satu unit yaitu Unit Proyek Pengembangan Bisnis yang bertujuan untuk menggenjot peningkatan pendapatan PD Kebersihan melalui pelayanan khusus, Bank Sampah Resik, Pelayanan *Cleaning Service, Advertising,* Pengadaan produk-produk Kebersihan dll.

2.1.4 Standar Operasional Prosedur Pengangkutan Sampah

Standar Operasional Prosedur (SOP) adalah dokumen yang berkaitan dengan prosedur yang dilakukan secara kronologis untuk menyelesaikan suatu pekerjaan yang bertujuan untuk memperoleh hasil kerja yang paling efektif dari para pekerja dengan biaya yang serendah-rendahnya. Adapun SOP dari kegiatan operasional pengangkutan sampah pada PD. Kebersihan adalah sebagai berikut.

SOP PENGANGKUTAN SAMPAH DARI TPS – TPA (MENGGUNAKAN *AMROLL TRUCK*)

No	Aktivitas	Petugas Pengangkutan	Kepala Urusan	Kepala Bidang Operasional	Mutu Baku		
					Persyaratan	Output	Waktu
1	Petugas pengangkutan menggunakan seragam dan menyiapkan perlengkapan pengangkutan	petugas melakukan persiapan pengangkutan			Peralatan: 1 Jaring 2 Terpal 3 Truk Amroll 4 Steel Container Seragam: 1 Pakaian Dinas Lapangan 2 Masker 3 Sarung tangan (Dikecualikan bagi supir) 4 Sepatu	Seragam dan peralatan kerja sesuai dengan standar	15 menit
2	Petugas pengangkutan mengecek kendaraan	pengecekan kendaraan			1 Oli 2 Air 3 Bahan bakar minyak 4 Ban 5 Lampu 6 Rem	Kendaraan dalam kondisi layak pakai	15 menit
3	Petugas menerima surat perintah jalan dari Kepala Bidang Operasional melalui Kepala urusan yang berisi: 1 Lokasi dan ritasi pengangkutan 2 Biaya bahan bakar minyak, biaya tol dan biaya urusan desa	surat perintah jalan dari kepala bidang operasional	surat perintah jalan	surat perintah jalan		1 Surat Perintah Jalan diterima petugas pengangkutan 2 Petugas memahami output pekerjaan	5 menit
4	Petugas berangkat dari pool untuk mengisi BBM ke SPBU yang ditunjuk	pengisian BBM			Surat Perintah Jalan	BBM terisi sesuai dengan indeks penetapan pemberian bahan bakar	45 menit
5	Petugas menuju TPS	menuju TPS					15 - 60 menit
6	Petugas menurunkan kontainer di TPS	pengangkutan sampah dari TPS				kontainer sampah siap menampung sampah yang baru	15 menit
7	Setelah steel Kontainer penuh terisi sampah, petugas menutup sampah dengan terpal dan jaring pengaman lalu diangkut dengan memperhatikan kapasitas kendaraan	menutup sampah di kendaraan dengan terpal dan jaring pengaman lalu mengangkut sampah				1.Sampah di TPS terangkut dan tidak ada ceceran sampah 2. sampah dalam kondisi aman dan siap diangkut	20 menit
8	Petugas membuang sampah ke TPA	membuang sampah ke TPA				Sampah terbuang ke TPA	2 - 4 jam
9	Petugas meminta bukti volume hasil penimbangan sampah yang dibuang ke TPA dari petugas TPA	bukti hasil penimbangan sampah dari TPA				Bukti hasil penimbangan sampah	10 menit
10	Petugas pengangkutan melaporkan hasil pekerjaannya kepada Kepala Bidang Operasional melalui Kepala Urusan dan Kendaraan kembali ke pool	pelaporan pada kepala bidang operasional dan mengembalikan kendaraan ke pool	laporan dari petugas pengangkutan	menerima laporan dari petugas pengangkutan	Dokumen yang dilaporkan: 1 Lokasi pengangkutan 2 Bon pengisian BBM, bon tol dan bon iuran desa 3 Bukti hasil penimbangan sampah		5 menit

Gambar 2.3 SOP Pengangkutan Sampah Menggunakan *Amroll Truck*

SOP PENGANGKUTAN SAMPAH DARI TITIK PENGUMPULAN KE TPA (MENGGUNAKAN *DUMP TRUCK*)

No	Aktivitas	Petugas Pengangkutan	Kepala Urusan	Kepala Bidang Operasional	Mutu Baku		
					Persyaratan	Output	Waktu
1	Petugas pengangkutan menggunakan seragam dan menyiapkan perlengkapan pengangkutan	petugas pengangkutan melakukan persiapan			Peralatan: 1 Sapu 2 Gacok 3 Singkup 4 Cerangka 5 Terpal 6 Jaring 7 Dump Truck Seragam: 1 Pakaian Dinas Lapangan 2 Masker 3 Sarung tangan 4 Sepatu	Seragam dan peralatan kerja sesuai dengan	15 menit
2	Petugas pengangkutan mengecek kendaraan	pengecekan kendaraan			1 Oli 2 Air 3 Bahan Bakar Minyak 4 Ban 5 Lampu 6 Rem	Kendaraan dalam kondisi layak pakai	15 menit
3	Petugas menerima surat perintah jalan dari Kepala Bidang Operasional melalui Kepala Urusan yang berisi: 1 Lokasi dan ritasi pengangkutan 2 Biaya bahan bakar minyak, biaya tol dan biaya urusan desa	surat perintah jalan dari kepala bidang operasional	surat perintah jalan	surat perintah jalan		1 Surat Perintah Jalan diterima petugas pengangkutan 2 Petugas memahami output pekerjaan	5 menit
4	Petugas berangkat dari pool untuk mengisi BBM ke SPBU yang ditunjuk	Pengisian BBM			Surat Perintah Jalan	BBM terisi sesuai dengan indeks penetapan	45 menit
5	Petugas menuju titik pengumpulan yang ditentukan	menuju titik pengumpulan					30 menit
6	Petugas mengangkut sampah dari titik pengumpulan dengan memperhatikan kapasitas kendaraan	pengangkutan sampah dari titik pengumpulan				Sampah pada titik pengumpulan terangkut dan tidak ada ceceran dan tumpukan sampah pada titik pengumpulan	
7	Setelah bak truk penuh terisi sampah, petugas menutup sampah pada kendaraan pengangkut dengan terpal dan jaring pengaman	menutup sampah di kendaraan dengan terpal dan jaring pengaman				Sampah dalam kondisi aman dan siap diangkut	20 menit
8	Petugas membuang sampah ke TPA	membuang sampah ke TPA				Sampah terbuang ke TPA	2 - 4 jam
9	Petugas meminta bukti volume hasil penimbangan sampah yang dibuang ke TPA dari petugas TPA	bukti hasil penimbangan sampah dari TPA				Bukti hasil penimbangan sampah	10 menit
10	Petugas kembali ke pool dengan melaporkan hasil pekerjaannya kepada Kepala Bidang Operasional melalui Kepala Urusan	kembalik ke pool dan melaporkan hasil pekerjaan pada kepala bidang operasional melalui Kepala Urusan	Laporan dari petugas pengakutan	menerima laporan dari petugas pengangkutan	Dokumen yang dilaporkan: 1 Lokasi pengangkutan 2 Bon pengisian BBM, bon tol dan bon 3 Bukti hasil penimbangan		5 menit

Gambar 2.4 SOP Pengangkutan Sampah Menggunakan *Dump Truck*

SOP PENGANGKUTAN SAMPAH DARI TITIK PENGUMPULAN KE TPA (MENGGUNAKAN *TRUCK COMPACTOR*)

No	Aktivitas	Petugas Pengangkutan	Kepala Urusan	Kepala Bidang Operasional	Mutu Baku		
					Persyaratan	Output	Waktu
1	Petugas pengangkutan menggunakan seragam dan menyiapkan perlengkapan pengangkutan	petugas pengangkutan melakukan persiapan			Peralatan: 1 Sapu 2 Gacok 3 Singkup 4 Cerangka 5 Truck Compactor Seragam: 1 Pakaian Dinas Lapangan 2 Masker 3 Sarung tangan (Dikecualikan bagi supir) 4 Sepatu	Seragam dan peralatan kerja sesuai dengan standar	15 menit
2	Petugas pengangkutan mengecek kendaraan	pengecekan kendaraan			1 Oli 2 Air 3 Bahan bakar 4 Ban 5 Lampu 6 Rem	Kendaraan dalam kondisi layak pakai	15 menit
3	Petugas menerima surat perintah jalan dari Kepala Bidang Operasional melalui Kepala Urusan yang berisi: 1 Lokasi dan ritasi pengangkutan 2 Biaya bahan bakar minyak, biaya tol dan biaya urusan desa	surat perintah jalan dari kepala bidang	surat perintah jalan	surat perintah jalan		1 Surat Perintah Jalan diterima petugas pengangkutan 2 Petugas memahami output pekerjaan	5 menit
4	Petugas berangkat dari pool untuk mengisi BBM ke SPBU yang ditunjuk	pengisian BBM			Surat Perintah Jalan	BBM terisi sesuai dengan indeks penetapan pemberian	45 menit
5	Petugas menuju titik pengumpulan yang ditentukan	menuju titik pengumpulan sampah					30 menit
6	Petugas mengangkut sampah dari titik pengumpulan yang ditentukan dengan memperhatikan kapasitas kendaraan	mengangkut sampah dari titik pengumpulan				Sampah pada titik pengumpulan terangkut dan tidak ada ceceran dan tumpukan sampah	2 - 4 jam
7	Petugas membuang sampah ke TPA	membuang sampah ke TPA				Sampah terbuang ke TPA	10 menit
8	Petugas meminta bukti volume hasil penimbangan sampah yang dibuang ke TPA dari petugas TPA	bukti hasil penimbangan sampah dari TPA				Bukti hasil penimbangan	5 menit
9	Petugas kembali ke pool dengan melaporkan hasil pekerjaannya kepada Kepala Bidang Operasional melalui Kepala Urusan	kembali ke pool dan melaporkan hasil pekerjaannya	laporan hasil pekerjaan	menerima laporan dari petugas pengangkutan	Dokumen yang dilaporkan: Lokasi 1 pengangkutan 2 Bon pengisian BBM, bon tol dan bon iuran desa 3 Bukti hasil penimbangan sampah		

Gambar 2.5 SOP Pengangkutan Sampah Menggunakan *Truck Compactor*

2.2 Landasan Teori

Landasan teori menjabarkan tentang istilah-istilah penting yang terdapat pada penelitian. Penjelasan secara singkat dipaparkan untuk mengetahui gambaran umum tentang istilah-istilah tersebut beserta kaitannya dengan penelitian.

2.2.1 Aplikasi

Aplikasi perangkat lunak adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas

yang diinginkan pengguna. Aplikasi perangkat lunak adalah program yang membuat komputer dapat digunakan untuk pekerjaan sehari-hari. Aplikasi perangkat lunak adalah program yang paling banyak digunakan oleh pengguna pada komputer [14]. Biasanya dibandingkan dengan perangkat lunak sistem yang mengintegrasikan berbagai kemampuan komputer tetapi tidak secara langsung menerapkan kemampuan tersebut untuk mengerjakan suatu tugas yang menguntungkan pengguna, aplikasi perangkat lunak berinteraksi dan dapat diarasakan kegunaanya secara langsung oleh pengguna.

Aplikasi perangkat lunak, atau hanya aplikasi, sering disebut 'program produktivitas' atau 'program *end-user*' karena memungkinkan pengguna untuk menyelesaikan tugas-tugas seperti membuat dokumen, spreadsheet, database, dan publikasi, melakukan riset online, mengirim email, membuat grafik , menjalankan bisnis, dan bahkan bermain game. Aplikasi perangkat lunak khusus untuk tugas dirancang untuk dan dapat sebagai sebagai aplikasi kalkulator yang sederhana atau serumit aplikasi pengolah kata.

2.2.2 Android

Android adalah sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware*, dan aplikasi [15]. Android awalnya dikembangkan oleh Android, Inc., dengan dukungan finansial dari Google, yang kemudian membelinya pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya Open Handset Alliance, konsorsium dari perusahaan-perusahaan perangkat keras, perangkat lunak, dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler.

Versi Android diawali dengan dirilisnya Android beta pada bulan November 2007. Versi komersial pertama, Android 1.0, dirilis pada September 2008. Android dikembangkan secara berkelanjutan oleh Google dan Open Handset Alliance (OHA), yang telah merilis sejumlah pembaruan sistem operasi ini sejak dirilisnya versi awal. Berikut adalah daftar versi android hingga bulan april 2018.

Tabel 2.1 Versi Android

Code Name	**Versi**	**Tanggal Rilis**	**API level**
(Tidak ada)	1.0	23 September 2008	1
(Disebut "Petit Four" di dalam perusahaan)	1.1	9 February 2009	2
Cupcake	1.5	27 April 2009	3
Donut	1.6	15 September 2009	4
Eclair	2.0 – 2.1	26 October 2009	5 – 7
Froyo	2.2 – 2.2.3	20 May 2010	8
Gingerbread	2.3 – 2.3.7	6 December 2010	9 – 10
Honeycomb	3.0 – 3.2.6	22 February 2011	11 – 13
Ice Cream Sandwich	4.0 – 4.0.4	18 Otober 2011	14 – 15
Jelly Bean	4.1 – 4.3.1	9 Juli 2012	16 – 18
KitKat	4.4 – 4.4.4	31 October 2013	19 – 20
Lollipop	5.0 – 5.1.1	12 November 2014	21 – 22
Marshmallow	6.0 – 6.0.1	5 Oktober 2015	23
Nougat	7.0 – 7.1.2	22 Agustus 2016	24 – 25
Oreo	8.0 – 8.1	21 Agustus2017	26 – 27

2.2.3 Web Service

Web service adalah sebuah aplikasi lintas platform yang dapat diakses melalui jaringan (intranet dan internet). Dimana dalam aplikasi tersebut menyediakan metode-metode dengan tujuan digunakan untuk interaksi aplikasi satu dengan aplikasi yang lain diakses dengan URL dan menerima response berbentuk JSON, XML, TXT, CSV dan lainnya.

2.2.4 Application Programming Interface (API)

Application Programming Interface yang biasa disingkat API adalah aplikasi yang memungkinkan developer untuk mengintegrasikan dua bagian dari aplikasi atau dengan aplikasi yang berbeda secara bersamaan. API terdiri dari

berbagai elemen seperti *function, protocols*, dan *tools* lainnya yang memungkinkan *developers* untuk membuat aplikasi. Tujuan penggunaan API adalah untuk mempercepat proses development dengan menyediakan function secara terpisah sehingga developer tidak perlu membuat fitur yang serupa. Penerapan API akan sangaat terasa jika fitur yang diinginkan sudah sangat kompleks, tentu membutuhkan waktu untuk membuat yang serupa dengannya.

API yang bekerja pada tingkat sistem operasi membantu aplikasi berkomunikasi dengan layer dasar dan satu sama lain mengikuti serangkaian protokol dan spesifikasi. Contoh yang dapat menggambarkan spesifikasi tersebut adalah POSIX (*Portable Operating System Interface*). Dengan menggunakan standar POSIX, aplikasi yang di-compile untuk bekerja pada sistem operasi tertentu juga dapat bekerja pada sistem lain yang memiliki kriteria yang sama. *Software library* juga memiliki peran penting dalam menciptakan compatibility antar sistem yang berbeda.

Aplikasi yang berinteraksi dengan *library* harus mengikuti serangkaian aturan yang ditentukan oleh *API*. Pendekatan ini memudahkan software developer untuk membuat aplikasi yang berkomunikasi dengan berbagai library tanpa harus memikirkan kembali strategi yang digunakan selama semua library mengikut *API* yang sama. Kelebihan lain dari metode ini menunjukkan betapa mudahnya menggunakan library yang sama dengan bahasa pemrograman yang berbeda.

2.2.5 JSON

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

JSON terbuat dari dua struktur:

a. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek *(object*), rekaman *(record*), struktur *(struct*), kamus (*dictionary*), tabel hash *(hash table*), daftar berkunci (*keyed list*), atau *associative array.*
b. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik *(array*), vektor (*vector),* daftar *(list*), atau urutan *(sequence*).

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

2.2.6 MySQL

MySQL merupakan sebuah perangkat lunak atau *software* sistem manajemen basis data SQL atau DBMS Multithread dan multi user. MySQl sebenarnya merupakan turunan dari salah satu konsep utama dalam database untuk pemilihan atau seleksi dan pemasukan data yang memungkinkan pengoperasian data dikerjakan secara mudah dan otomatis. MySQL diciptakan oleh Michael "Monty" Widenius pada tahun 1979, seorang programmer komputer asal Swedia yang mengembangkan sebuah sistem database sederhana yang dinamakan UNIREG yang menggunakan koneksi low-level ISAM database engine dengan indexing.

Adapun kelebihan MySQl dalam penggunaanya dalam database adalah sebagai berikut.

a. *Free* atau gratis sehingga MySQL dapat dengan mudah untuk mendapatkannya.
b. MySQl stabil dan tangguh dalam pengoperasiannya
c. My SQl mempunyai sistem keamanan yang cukup baik
d. Sangat mendukung transaksi dan mempunyai banyak dukungan dari komunitas
e. Sangat fleksibel dengan berbagai macam program
f. Perkembangan dari MySQl sangat cepat

Selain kelebihan yang disampaikan diatas, ada beberapa kekurangan yang dimiliki oleh mySQl, diantaranya:

a. Kurang mendukung koneksi bahasa pemrograman seperti Visual basic atau biasa kita kenal dengan sebutan VB, Foxpro, Delphi dan lain-lain sebab koneksi ini menyebabkan *field* yang dibaca harus sesuai dengan koneksi dari bahasa pemrograman visual tersebut.
b. Data yang dapat ditangani belum besar dan belum mendukung widowing function.

2.3 Teknologi

Teknologi dalah sarana yang digunakan dalam pengoperasian aplikasi yang dibangun dalam penelitian ini. Penggunaan teknologi dimaksudkan untuk mempermudah dalam pengembangan dan juga dalam penggunaan aplikasi.

2.3.1 Google Maps API

Google Maps API adalah layanan untuk menambahkan peta berdasarkan data Google Maps. API secara otomatis menangani akses ke *server* Google Maps, pengunduhan data, tampilan peta, dan respons terhadap gerakan peta. Pengguna dapat menggunakan panggilan API untuk menambahkan penanda, poligon, dan hamparan ke peta dasar, dan untuk mengubah pandangan pengguna dari area peta tertentu. Objek-objek ini memberikan informasi tambahan untuk lokasi peta, dan memungkinkan interaksi pengguna dengan peta. API memungkinkan developer untuk menambahkan grafik berikut ke peta.

a) Ikon berlabuh ke posisi tertentu di peta (Marker).
b) Set segmen garis (Polylines).
c) Segmen tertutup (Polygons).
d) Grafik bitmap yang dilekatkan pada posisi tertentu di peta (Hamparan Tanah).
e) Set gambar yang ditampilkan di atas ubin peta dasar (Tile Overlay).

2.3.2 Google Maps Directions API

Google Maps Directions API adalah layanan yang dapat menghitung arah antar lokasi. Pengguna bisa menelusuri arah untuk beberapa moda transportasi, termasuk angkutan umum, mengemudi, berjalan atau bersepeda. Arah dapat menentukan posisi asal, tujuan maupun titik jalan baik sebagai *string teks* (misalnya "dago" atau "jl. dago no. 100, bandung, jawa barat") atau sebagai koordinat garis bujur/garis lintang. Google Maps Directions API dapat membalikkan arah multi-bagian memakai serangkaian titik jalan. Fasilitas ini biasanya dirancang untuk mengukur arah alamat statis (yang sudah diketahui sebelumnya) untuk peletakan konten aplikasi di dalam peta, layanan ini tidak dirancang untuk merespon masukan *user* secara *real-time*.

1. Penggunaan Google Api Direction

   Untuk memakai fasilitas Google Api Direction bisa mengakses dengan permintaan HTTP. Permintaan Google Maps Directions API dilakukan pada alamat: https://maps.googleapis.com/maps/api/directions/output?parameters. Dalam hal ini, keluaran dapat berupa format XML atau JSON.

2. Parameter

   Untuk mengakses Google Api Direction menggunakan HTTP, parameter dibagi menjadi dua yaitu pertama parameter wajib yang wajib dibutuhkan, serta parameter tambahan. Parameter wajib yang dibutuhkan antara lain:

a. Origin

   Origin merupakan parameter untuk lokasi asal, parameter bisa berupa format String dengan alamat atau *Longitude Latitude*.

b. Destination

   Destination merupakan parameter untuk lokasi tujuan, parameter bisa berupa format String dengan alamat atau *Longitude Latitude*.

c. Key

   Key merupakan parameter yang digunakan untuk mengidentifikasi pengakses layanan.

Adapun parameter opsional adalah sebagai berikut.

a) mode (default-nya adalah *driving*)

Menetapkan moda transportasi yang akan digunakan saat menghitung arah.

b) Waypoint

Menetapkan larik titik jalan.

c) Alternative

Jika disetel ke *true*, menetapkan bahwa layanan Directions mungkin menyediakan lebih dari satu rute alternatif dalam respons.

d) Avoid

Menunjukkan rute yang dihitung harus menghindari fitur yang ditandai. Parameter ini mendukung argumen berikut.

e) Language

Bahasa yang digunakan untuk mengembalikan hasil.

f) Unit

Menetapkan sistem satuan yang akan digunakan saat menampilkan hasil.

g) Region

Menetapkan kode region, ditetapkan sebagai nilai yang berisi dua karakter.

h) arrival_time

Menetapkan waktu kedatangan yang diinginkan untuk arah angkutan umum.

i) departure_time

Menetapkan waktu keberangkatan yang diinginkan untuk kedaraan umum.

j) Untuk permintaan yang mode perjalanannya adalah mengemudi: Anda bisa menetapkan departure_time untuk menerima durasi rute dan perjalanan (bidang respons*: duration_in_traffic*) yang mempertimbangkan kondisi lalu lintas. Opsi ini hanya tersedia jika permintaan berisi kunci API yang valid, atau tanda tangan dan ID klien Google Maps APIs Premium Plan yang valid. departure_time harus disetel ke waktu saat ini atau waktu mendatang. Tidak boleh waktu yang sudah lewat.

k) *traffic_model* (default-nya adalah best_guess)

Menetapkan asumsi yang akan digunakan saat menghitung waktu dalam lalu lintas.

2.3.3 Geofence

Geofence adalah *virtual perimeter* untuk area geografis dunia nyata. *Geofence* dapat dihasilkan secara dinamis, seperti dalam radius di sekitar lokasi titik, atau *geo-fence* dapat menjadi seperangkat batas yang telah ditetapkan (seperti zona sekolah atau batas lingkungan).

Penggunaan geo-fence disebut *geo-fencing*, dan satu contoh penggunaan melibatkan perangkat yang sadar lokasi dari pengguna layanan berbasis lokasi (LBS) yang masuk atau keluar dari *geo-fence*. Aktivitas ini dapat memicu peringatan bagi pengguna perangkat serta pengiriman pesan ke operator *geo-fence*. Info ini, yang dapat berisi lokasi perangkat, dapat dikirim ke telepon seluler atau akun email. Penggunaan *geo-fence* pada penelitian ini yaitu untuk membuat *geo-fence* pada saat *driver* telah dekat ke lokasi pengangkutan sampah sehingga petugas dapat menerima notifikasi kedatangan *driver*.

2.3.4 Firebase Cloud Messaging

Firebase Cloud Messaging (FCM), yang sebelumnya dikenal sebagai *Google Cloud Messaging* (GCM), adalah solusi lintas platform untuk pesan dan notifikasi untuk Android, iOS, dan aplikasi web, yang saat ini dapat digunakan tanpa biaya. Adapun fitur utama FCM adalah sebagai berikut.

a. **Mengirim pesan notification atau pesan data**. Mengirim pesan notifikasi yang ditampilkan kepada pengguna. Atau mengirim pesan data dan menentukan sepenuhnya apa yang terjadi dalam kode aplikasi.
b. **Penargetan pesan serbaguna**. Mendistribusikan pesan ke aplikasi klien dengan salah satu dari 3 cara: ke satu perangkat, ke grup perangkat, atau ke perangkat yang berlangganan topik.
c. **Mengirim pesan dari aplikasi klien**. Mengirim notifikasi, *chat*, dan pesan lain dari perangkat ke server melalui saluran koneksi FCM yang andal dan hemat baterai.

FCM dapat mengirim 2 jenis pesan ke klien yaitu sebagai berikut.

1 Pesan notifikasi, terkadang dianggap sebagai "pesan tampilan". Pesan ini ditangani oleh FCM SDK secara otomatis.

2 Pesan data, yang ditangani oleh aplikasi klien

Pesan notifikasi memiliki serangkaian kunci bawaan yang terlihat oleh pengguna. Sebaliknya, pesan data hanya memuat key-value pair kustom buatan pengguna. Pesan notifikasi bisa berisi payload data opsional. Payload maksimum untuk kedua jenis pesan tersebut berukuran 4KB, kecuali saat mengirim pesan dari Firebase console, yang memberlakukan batas 1.024 karakter.

Tabel 2.2 Perandingan Pesan Notifikasi dan Pesan Data

	Skenario penggunaan	Cara mengirim
Pesan Notifikasi	FCM secara otomatis menampilkan pesan ke perangkat pengguna akhir atas nama aplikasi klien. Pesan notifikasi memiliki serangkaian kunci bawaan yang terlihat oleh pengguna dan payload data opsional untuk key-value pair kustom.	1. Dalam lingkungan tepercaya, seperti Cloud Functions atau server aplikasi Anda, gunakan Admin SDK atau Protokol Server FCM: Setel kunci notification. Dapat memiliki payload data opsional. Selalu dapat diciutkan. 2. Gunakan Notifications composer: Masukkan Teks Pesan, Judul, dll., lalu kirim. Tambahkan payload data opsional dengan memasukkan Data kustom
Pesan Data	Aplikasi klien bertanggung jawab memproses pesan data. Pesan data hanya memuat key-value pair kustom.	Dalam lingkungan tepercaya, seperti Cloud Functions atau server aplikasi Anda, gunakan Admin SDK atau Protokol Server FCM: Setel kunci data saja.

2.3.5 QR Code

QR Code singkatan dari *Quick Response Code* adalah jenis *barcode* yang berisi matriks dots yang dapat dipindai/scan menggunakan QR scanner atau

smartphone dengan built-in kamera. Setelah dipindai/scan, perangkat lunak pada QR scanner atau *smarphone* akan mengubah kode titik-titik ke dalam karakter angka maupun string. Misalnya, memindai QR Code dengan *smartphone* mungkin akan membuka URL di *web browser smartphone*.

Semua QR Code memiliki bentuk persegi dan mencakup tiga garis persegi di bagian bawah-kiri, atas-kiri, dan sudut kanan atas. Garis persegi menentukan orientasi kode. Titik-titik dalam QR Code mengandung Format dan informasi versi serta konten itu sendiri. QR Code juga mencakup koreksi kesalahan tingkat tertentu , yang didefinisikan sebagai L, M, Q, atau H. Sejumlah koreksi kesalahan rendah (L) memungkinkan QR Code mengandung lebih banyak konten, sedangkan koreksi kesalahan yang lebih tinggi (H) membuat kode lebih mudah untuk memindai.

Keuntungan lain dari QR Code adalah bahwa mereka dapat dipindai dari layar. Standar scanner UPC menggunakan laser untuk memindai barcode, yang berarti tidak dapat memindai UPC dari layar. QR scanner, dirancang untuk bisa menangkap gambar 2D yang dicetak di atas kertas atau yang ditampilkan di sebuah layar.

2.4 Bahasa Pemrograman

Bahasa Pemrograman adalah sebuah instruksi standar untuk memerintah komputer agar menjalankan fungsi tertentu. Bahasa pemrograman ini merupakan suatu himpunan dari aturan sintaks dan semantik yang dipakai untuk mendefinisikan program komputer. Bahasa ini memungkinkan seorang *programmer* dapat menentukan secara persis data mana yang akan diolah oleh komputer, bagaimana data ini akan disimpan/diteruskan, dan jenis langkah apa secara persis yang akan diambil dalam berbagai situasi.

2.4.1 Java

Java adalah bahasa pemrograman yang dapat dijalankan dimanapun dan di sembarang platform apapun di beragam lingkungan: *internet, intranet, comsumer electronic products, dan computer applications* [16]. Bahasa ini awalnya dibuat oleh James Gosling saat masih bergabung di Sun Microsystems saat ini merupakan

bagian dari Oracle dan dirilis tahun 1995. Bahasa ini banyak mengadopsi sintaksis yang terdapat pada C dan C++ namun dengan sintaksis model objek yang lebih sederhana serta dukungan rutin-rutin aras bawah yang minimal. Aplikasi berbasis java pada umumnya di *compile* menjadi p-code (bytecode) dan dapat berjalan pada beragam *Java Virtual Machine* (JVM) [17].

Aplikasi-aplikasi berbasis java umumnya dikompilasi ke dalam p-code (bytecode) dan dapat dijalankan pada berbagai Mesin Virtual Java (JVM). Java merupakan bahasa pemrograman yang bersifat umum/non-spesifik (general purpose), dan secara khusus didisain untuk memanfaatkan dependensi implementasi seminimal mungkin. Karena fungsionalitasnya yang memungkinkan aplikasi java mampu berjalan di beberapa platform sistem operasi yang berbeda, java dikenal pula dengan slogannya, "Tulis sekali, jalankan di mana pun". Saat ini java merupakan bahasa pemrograman yang paling populer digunakan, dan secara luas dimanfaatkan dalam pengembangan berbagai jenis perangkat lunak aplikasi ataupun aplikasi.

2.4.2 PHP

PHP dikenal secara umum dikenal sebagai bahasa pemrograman *script-script* yang membuat dokumen HTML secara *on the fly* yang dieksekusi di server web, dokumen yang dihasilkan dari suatu aplikasi bukan dokumen HTML yang dibuat dengan menggunakan editor teks atau editor HTML, dikenal juga sebagai bahasa pemrograman server side [18]. PHP diciptakan oleh Rasmus Lerdorf pertama kali tahun 1994. Pada awalnya PHP adalah singkatan dari "Personal Home Page Tools". Selanjutnya diganti menjadi FI ("Forms Interpreter"). Sejak versi 3.0, nama bahasa ini diubah menjadi "PHP: Hypertext Prepocessor" dengan singkatannya "PHP".

Beberapa kelebihan PHP sebagai bahasa pemrograman web antara lain sebagai berikut.

1. Bahasa pemrograman PHP adalah sebuah bahasa script yang tidak melakukan sebuah kompilasi dalam penggunaannya.

2. *Web Server* yang mendukung PHP dapat ditemukan di mana - mana dari mulai apache, IIS, Lighttpd, hingga Xitami dengan konfigurasi yang relatif mudah.
3. Dalam sisi pengembangan lebih mudah, karena banyaknya milis - milis dan developer yang siap membantu dalam pengembangan.
4. Dalam sisi pemahamanan, PHP adalah bahasa scripting yang paling mudah karena memiliki referensi yang banyak.
5. PHP adalah bahasa open source yang dapat digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara runtime melalui console serta juga dapat menjalankan perintah-perintah system.

2.4.3 SQL

SQL (*Structured Query Language*) adalah sebuah bahasa yang digunakan untuk mengakses data dalam basis data relasional. Bahasa ini merupakan bahasa standar yang digunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya. SQL adalah bahasa generasi ke-empat (4GL) yang dibuat menjadi mudah dan cepat dipelajari. SQL diciptakan oleh Dr. E.F. Codd dan IBM pada awal 1970, yang kemudian ANSI (American National Standards Institute) mengakuinya dan mempulikasikannya sebagai bahasa standar [19].

Secara umum, SQL terdiri dari dua bahasa, yaitu *Data Definition Language* (DDL) dan *Data Manipulation Language* (DML). Implementasi DDL dan DML berbeda untuk tiap sistem manajemen basis data (SMBD), namun secara umum implementasi tiap bahasa ini memiliki bentuk standar yang telah ditetapkan.

DDL digunakan untuk mendefinisikan, mengubah, serta menghapus basis data dan objek-objek yang diperlukan dalam basis data, misalnya tabel, view, user, dan sebagainya. Secara umum, DDL yang digunakan adalah *CREATE* untuk membuat objek baru, *USE* untuk menggunakan objek, *ALTER* untuk mengubah objek yang sudah ada, dan *DROP* untuk menghapus objek. DDL biasanya digunakan oleh administrator basis data dalam pembuatan sebuah aplikasi basis data.

DML digunakan untuk memanipulasi data yang ada dalam suatu tabel. Berikut adalah perintah yang umum dilakukan pada DML.

a. *SELECT* untuk menampilkan data.
b. *INSERT* untuk menambahkan data baru.
c. *UPDATE* untuk mengubah data yang sudah ada.
d. *DELETE* untuk menghapus data.

2.5 Perangkat Lunak Pendukung

Terdapat beberapa perangkat lunak yang digunakan dalam pembangunan aplikasi ini. Perangkat lunak tersebut yaitu Android Studio, Sublime Text, dan XAMPP.

2.5.1 Android Studio

Android Studio adalah lingkungan pengembangan terintegrasi (IDE) resmi untuk sistem operasi Android dari Google, yang dibangun di perangkat lunak IntelliJ IDEA JetBrains dan dirancang khusus untuk pengembangan Android. Perangkat lunak ini adalah pengganti Eclipse Android Development Tools (ADT) sebagai IDE utama untuk pengembangan aplikasi Android asli.

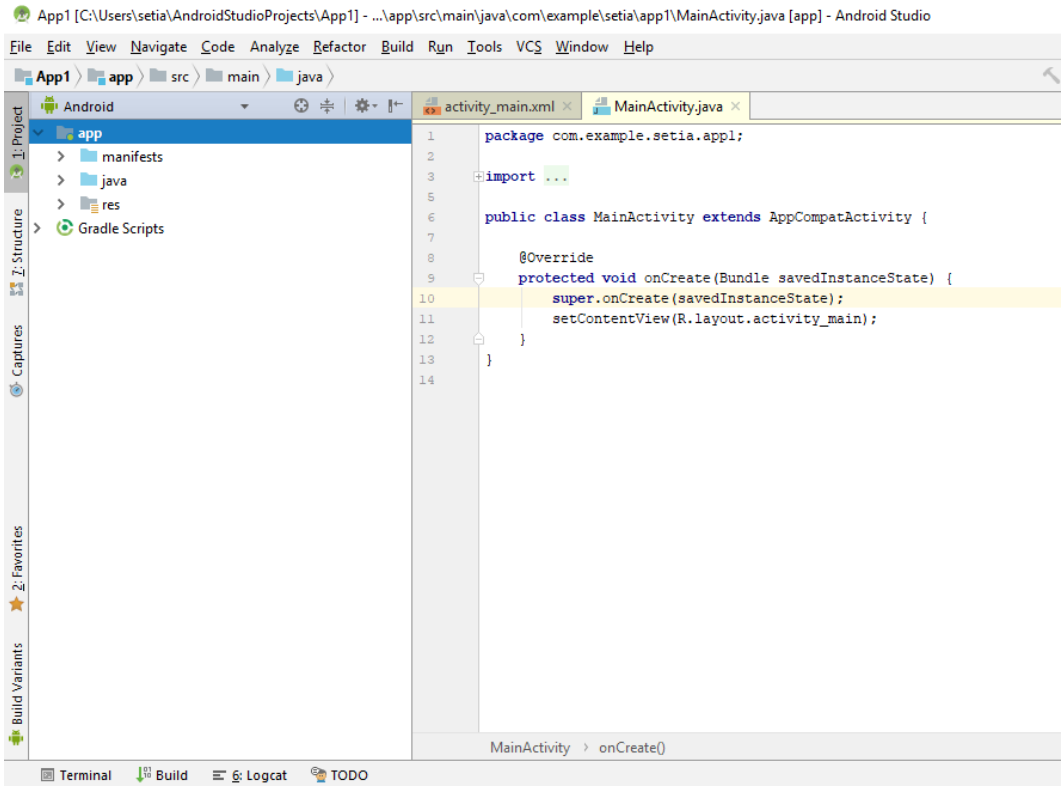
Android Studio mendukung semua bahasa pemrograman yang sama dari IntelliJ, dan PyCharm misalnya Python, dan Kotlin, dan Android Studio 3.0 mendukung fitur bahasa Java 7 dan subset fitur bahasa Java 8 yang bervariasi menurut versi platform. Proyek eksternal mendukung beberapa fitur Java 9.

Berikut adalah fitur-fitur yang tersedia pada android studio.

a. Dukungan build berbasis Gradle.
b. Refactoring khusus dan perbaikan cepat Android.
c. Alat serat untuk menangkap kinerja, kegunaan, kompatibilitas versi, dan masalah lainnya.
d. Integrasi ProGuard dan kemampuan penandatanganan aplikasi
e. *Wizard* berbasis template untuk membuat desain dan komponen Android yang umum.

f. Editor tata letak kaya yang memungkinkan pengguna untuk menyeret dan menjatuhkan komponen UI, opsi untuk melihat tata letak pada beberapa konfigurasi layar.
g. Dukungan untuk membangun aplikasi Android Wear.
h. Dukungan bawaan untuk Google Cloud Platform, memungkinkan integrasi dengan *Firebase Cloud Messaging* ('Perpesanan Google Cloud' Sebelumnya) dan Google App Engine.
i. Android Virtual Device (Emulator) untuk menjalankan dan men-debug aplikasi di studio Android.

Tampilan Android Studio dapat dilihat pada gambar berikut.



Gambar 2.6 Tampilan Android Studio

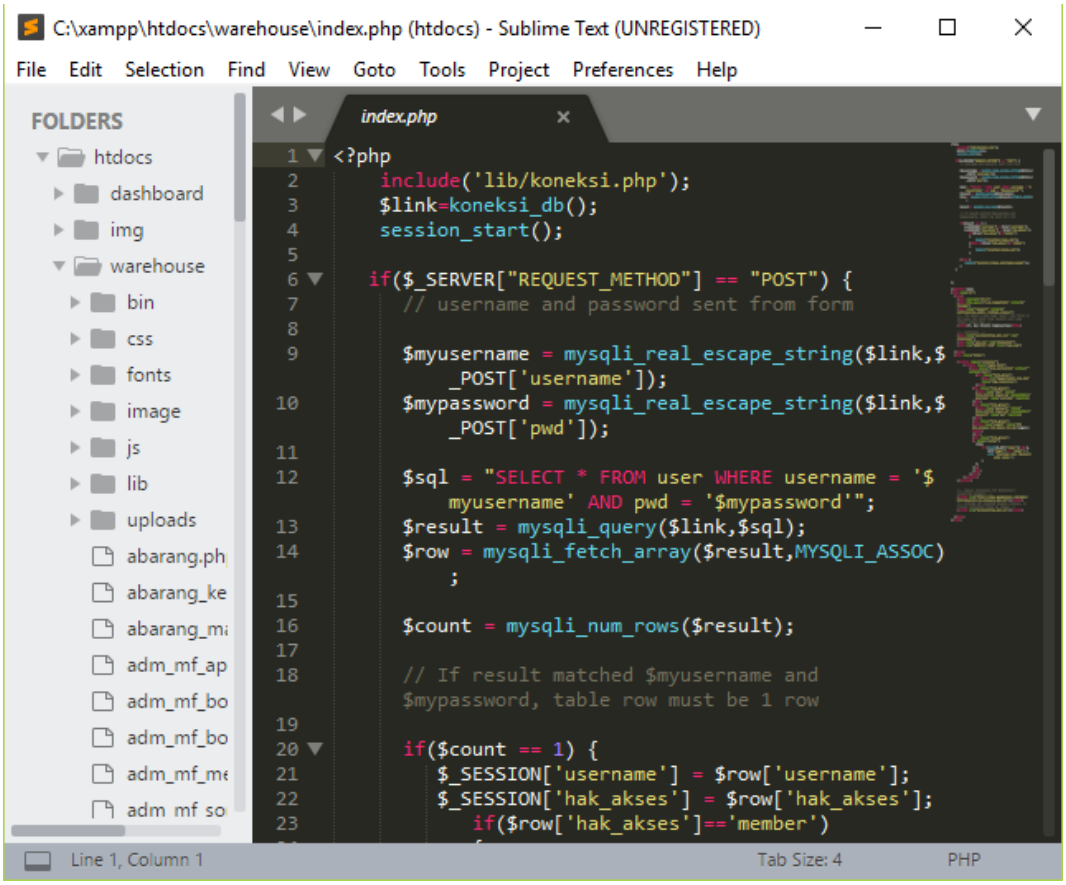
2.5.2 Sublime Text

Sublime Text adalah editor *source code* lintas platform dengan *Application Programming Interface* (API) Python. Sublime Text secara nativ mendukung banyak bahasa pemrograman dan bahasa markup, dan fungsi dapat ditambahkan

oleh pengguna dengan plugin, biasanya dibuat oleh komunitas dan dipelihara di bawah lisensi perangkat lunak bebas. Berikut adalah daftar fitur yang terdapat Sublime Text.

a. "*Goto Anything*," navigasi cepat ke file, simbol, atau garis.
b. "*Command palette*" menggunakan pencocokan adaptif untuk permintaan keyboard cepat dari perintah asal.
c. Pengeditan simultan: secara bersamaan melakukan perubahan interaktif yang sama ke beberapa area yang dipilih.
d. API plugin berbasis Python.
e. Preferensi khusus project.
f. Ekstensif *customizability* melalui file pengaturan JSON, termasuk pengaturan spesifik proyek dan platform-spesifik
g. *Cross-platform* (Windows, macOS, dan Linux) dan Plugin Pendukung untuk *cross-platform.*
h. Kompatibel dengan banyak grammar bahasa dari TextMate.

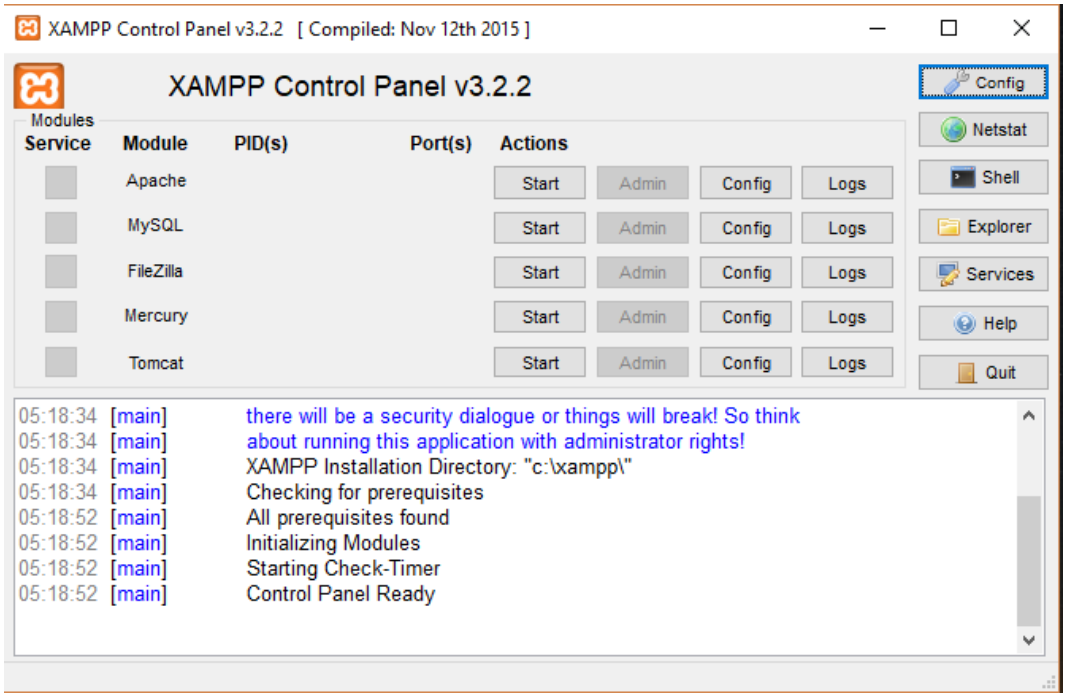
Tampilan Sublime Text dapat dilihat pada gambar berikut.



Gambar 2.7 Tampilan Sublime Text

2.5.3 XAMPP

XAMPP adalah perangkat lunak bebas yang berfungsi sebagai server yang berdiri sendiri (localhost), yang terdiri atas program Apache HTTP Server, MySQL database, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan Perl. Nama XAMPP merupakan singkatan dari X (empat sistem operasi apapun), Apache, MySQL, PHP dan Perl. Program ini tersedia dalam GNU General Public License dan bebas, merupakan web server yang mudah digunakan yang dapat melayani tampilan halaman web yang dinamis. XAMPP secara reguler diperbarui ke rilis terbaru dari Apache, MariaDB, PHP dan Perl. XAMPP juga dilengkapi dengan sejumlah modul lain termasuk OpenSSL, phpMyAdmin, MediaWiki, Joomla, WordPress dan banyak lagi. Gambar berikut adalah contoh tampilan dari XAMPP.



Gambar 2.8 Tampilan XAMPP

2.6 Analisis dan Perancangan Berorientasi Objek

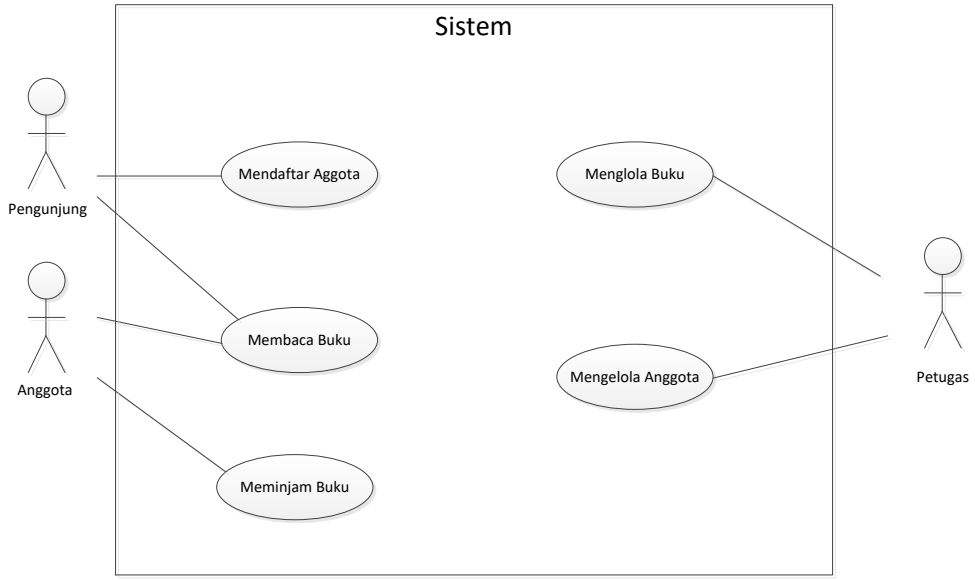
Analisis dan Perancangan Berorientasi Objek (OOAD) adalah rekayasa pendekatan perangkat lunak yang model sistemnya sebagai sekelompok objek yang saling berinteraksi. Setiap objek terdiri dari beberapa entitas yang mempunyai kepentingan dalam sistem yang dimodelkan, dan ditandai oleh class-classnya, elemen data dan perilakunya. Berbagai model dapat dibuat untuk menunjukkan struktur yang bersifat statis, perilaku dinamis dan run-time penyebaran objek yang berkolaborasi. Ada beberapa notasi yang berbeda untuk mewakili model ini, seperti Unified Modeling Language (UML).

UML (*Unified Modeling Language*) adalah sebuah bahasa untuk menentukan, visualisasi, kontruksi, dan mendokumentasikan *artifact* (bagian dari informasi yang digunakan atau dihasilkan dalam suatu proses pembuatan perangkat lunak. *Artifact* dapat berupa model, deskripsi atau perangkat lunak) dari sistem perangkat lunak, seperti pada pemodelan bisnis dan system non perangkat lunak lainnya.

Sebagai bahasa, UML dapat digunakan untuk mendeskripsikan sistem informasi yang dikembangkan menggunakan paradigm tradisional atau paradigm berbasis objek, termasuk penggabungan proses. UML adalah notasi yang dapat digunakan sebagai penghubung dengan berbagai metodologi [20].

2.6.1 Diagram *Use Case*

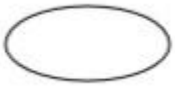
Use Case memodelkan interaksi antara aktor (pengguna exkternal dari sebuah sistem informasi) dengan sistem informasi itu sendiri. Diagram *Use Case* adalah diagram yang menggabungkan beberapa *use case* [20]. Berikut adalah contoh diagram *use case* perpustakaan.



Gambar 2.9 Contoh Diagram *Use Case*

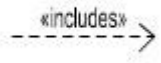
a. Element - elemen pada *Use Case Diagram*

Aktor : Mempresentasikan seseorang atau sesuatu(seperti perangkat,sistem lain) yang berinteraksi dengan sistem.*Actor* hanya berinteraksi dengan use case tetapi tidak memiliki kontrol atas use case.



***Use Case* :** Adalah gambaran fungsionalitas dari suatu sistem, sehingga customer atau pengguna sistem paham dan mengerti mengenai kegunaan sistem yang akan dibangun.

Asosiasi : Menghubungkan link antar element.



<<**Include**>> **:** Yaitu kelakuan yang harus terpenuhi agar sebuah event dapat terjadi, dimana pada kondisi ini sebuah *use case* adalah bagian dari *use case* lainnya.

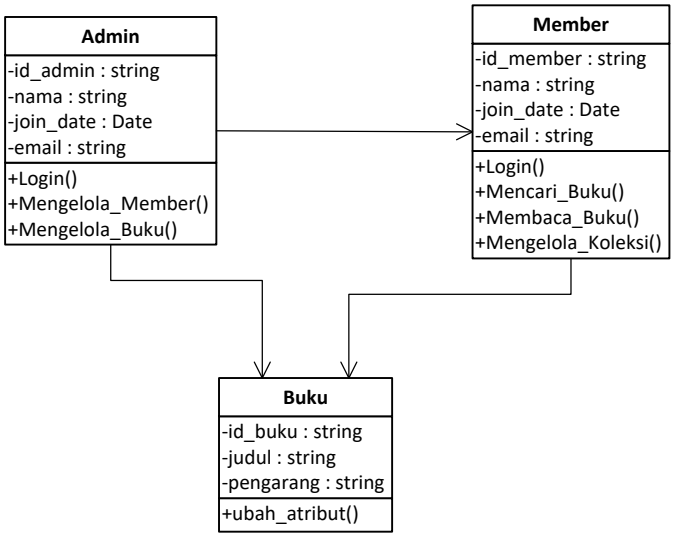
b. Relasi Dalam *Use Case*

Ada beberapa relasi yang terdapat pada diagram use case sebagai berikut.

1. Asosiasi, menghubungkan link antar element.
2. Generalisasi, disebut juga *inheritance* (pewarisan), sebuah elemen dapat merupakan spesialisasi dari elemen lainnya.
3. Dependensi, sebuah element bergantung dalam beberapa cara ke element lainnya.
4. Agregasi, bentuk assosiation dimana sebuah elemen berisi elemen lainnya.

2.6.2 Diagram *Class*

Diagram *Class* adalah model dari beberapa kelas yang menggambarkan hubungan statis diantara mereka, termasuk asosiasi dan generalisasi [20]. *Class* memiliki 3 area pokok (utama) yaitu: nama, atribut, dan operasi. Nama berfungsi untuk member identitas pada sebuah kelas, atribut fungsinya adalah untuk member karakteristik pada data yang dimiliki suatu objek di dalam kelas, sedangkan operasi fungsinya adalah memberikan sebuah fungsi ke sebuah objek. Gambar 2.10 berikut menunjukkan contoh diagram *class*.



Gambar 2.10 Contoh Diagram *Class*

Dalam mendefinisikan metode yang ada di dalam kelas harus diperhatikan yang namanya *Cohesion* dan *Coupling*, Cohesion adalah ukuran keterkaitan sebuah instruksi di sebuah metode, *Coupling* adalah ukuran keterkaitan antar metode. Di dalam class diagram terdapat hubungan antar kelas secara konseptual, yang disebut Relasi antar *Class*. Pada UML disediakan macam-macam relasi antar *Class*, diantaranya: Asosiasi (Hubungan statis antar kelas), Agregasi (hubungan dari keseluruhan objek), Generalisasi (relasi beberapa subkelas ke super kelas), *Dependency* (keterhubungan tiap kelas.)

Atribut dan metoda dapat memiliki salah satu sifat berikut :

c. *Private*, tidak dapat dipanggil dari luar *class* yang bersangkutan
d. *Protected*, hanya dapat dipanggil oleh *class* yang bersangkutan dan anak-anak yang mewarisinya
e. *Public*, dapat dipanggil oleh siapa saja
f. Hubungan Antar *Class:*
g. Asosiasi, yaitu hubungan statis antar class. Umumnya menggambarkan *class* yang memiliki atribut berupa class lain, atau class yang harus mengetahui eksistensi class lain. Panah *navigability* menunjukkan arah *query* antar class.
h. Agregasi, yaitu hubungan yang menyatakan bagian (“terdiri atas”).

i. Pewarisan, yaitu hubungan hirarki antar *class*. *Class* dapat diturunkan dari *class* lain dan mewarisi semua atribut dan metoda class asalnya dan menambahkan fungsionalitas baru, sehingga ia disebut anak dari *class* yang diwarisinya. Kebalikan dari pewarisan adalah generalisasi.
j. Hubungan dinamis, yaitu rangkaian pesan *(message)* yang di-passing dari satu class kepada *class* lain. Hubungan dinamis dapat digambarkan dengan menggunakan *sequence* diagram yang akan dijelaskan kemudian.

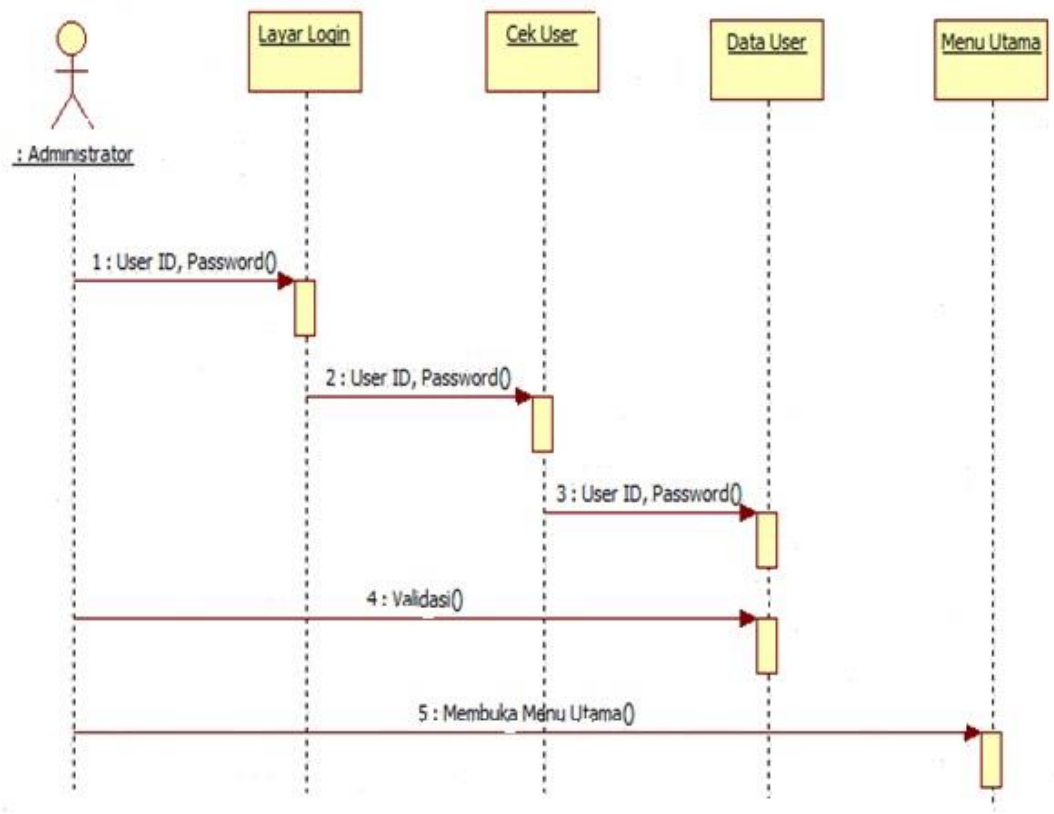
2.6.3 Diagram *Sequence*

Diagram *Sequence* menggambarkan objek berinteraksi satu sama lain dimana pesan dikirimkan diantara mereka. Diagram *sequence* juga merupakan model dinamis yang menggambarkan perilaku objek [20].

Berikut komponen - komponen yang ada pada sequence diagram.

a. *Object* adalah komponen berbentuk kotak yang mewakili sebuah class atau object. Mereka mendemonstrasikan bagaimana sebuah *object* berperilaku pada sebuah sistem.
b. *Activation boxes* - adalah komponen yang berbentuk persegi panjang yang menggambarkan waktu yang diperlukan sebuah object untuk menyelesaikan tugas. Lebih lama waktu yang diperlukan, maka *activation boxes* akan lebih panjang.
c. Aktor - adalah komponen yang berbentuk *stick figure*. Komponen yang mewakili seorang pengguna yang berinteraksi dengan sistem.
d. *Lifeline* - adalah komponen yang berbentuk garis putus - putus. *Lifeline* biasanya memuat kotak yang berisi nama dari sebuah *object*. Berfungsi menggambarkan aktifitas dari *object.*

Contoh diagram *sequence* dapat dilihat pada gambar 2.11 berikut.



Gambar 2.11 Contoh Diagram *Sequence*

2.6.4 Diagram *Activity*

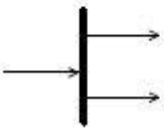
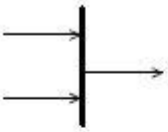
Diagram *Activity* menggambarkan event yang terjadi dalam waktu yang sama dikoordinasikan. Diagram *activity* juga merupakan model dinamis [20].

a. Fungsi Diagram *Activity*

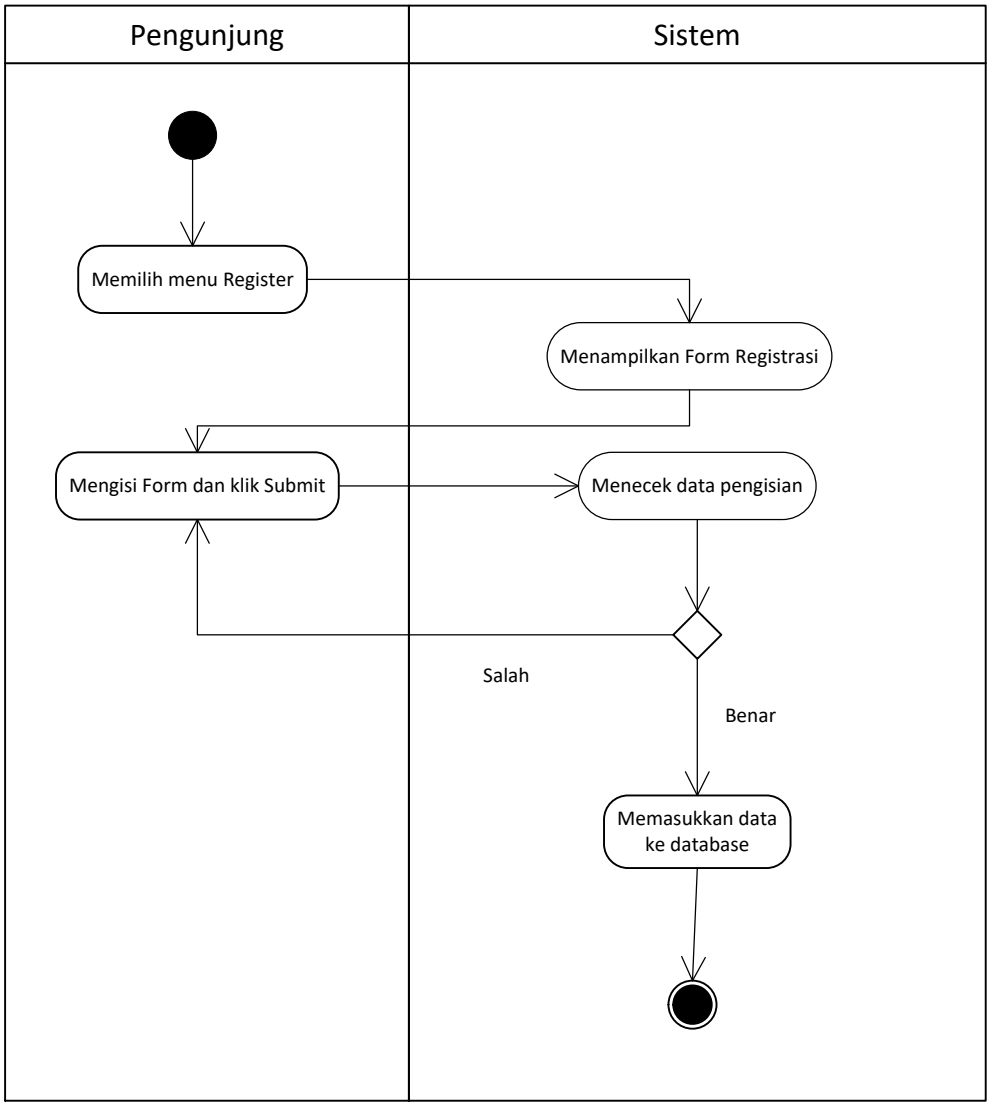
1. Menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses
2. Memperlihatkan urutan aktifitas proses pada sistem
3. *Activity diagram* dibuat berdasarkan sebuah atau beberapa *use case* pada *use case* diagram

b. Elemen - Elemen Pada *Activity* Diagram

Tabel 2.3 Elemen-Elemen Pada Diagram *Activity*

Simbol	Keterangan
	Start Point
	End Point
	Activities
	Fork (Percabangan)
	Join (Penggabungan)
	Decision
Swimlane	Sebuah cara untuk mengelompokkan activity berdasarkan Actor (mengelompokkan activity dalam sebuah urutan yang sama)

Gambar 2.12 berikut menunjukkan contoh sebuah diagram *activity*.



Gambar 2.12 Contoh Diagram *Activity*