BAB II
TINJAUAN PUSTAKA

II.1 Ruang Lingkup Objek Penelitian

Objek penelitian yang sedang siteliti adalah SMP Negeri 28 Bandung, yang bertempat di Jl. Solontongan No.2 Kota Bandung.

II.1.1 Profil SMP Negeri 28 Bandung

1. Nama Sekolah : SMP Negeri 28 Bandung
2. No. NSS : 20.1.02.60.13.126
3. No. NPSN : 20219466
   a. Alamat : Jl. Solontongan II Bandung
   b. Desa/Kelurahan : Turangga
   c. Kecamatan : Lengkong
4. Nama Kepala Sekolah : Yaya Supiyana, S.Pd., M.M.Pd.
5. No. Telp/HP : (022) 7304544 / 08122314038
6. Kategori Sekolah : SSN
   a. Tahun Didirikan/Th. Beroprasi : 1982
   b. Kepemilikan Tanah/Bangunan : Milik Pemerintah
   c. Luas Tanah / Status : 3930 m^2 Hak Pakai
7. Luas Bangunan : 3071 m^2

II.1.2 Logo SMP Negeri 28 Bandung

Berikut ini merupakan logo dari SMP Negeri 28 Bandung:



Gambar II.1 Logo SMP Negeri 28 Bandung

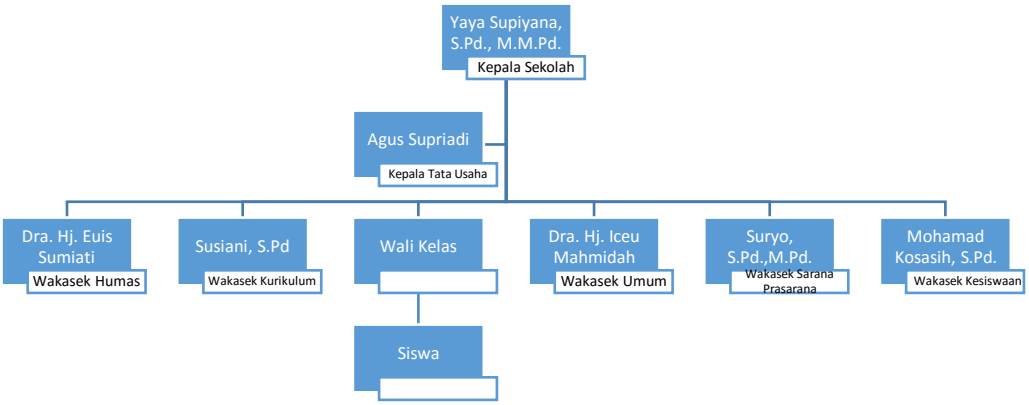
II.1.3 Visi dan Misi

1. Visi

Terwujudnya SMP Negeri 28 Bandung yang unggul dalam prestasi, tangguh dalam mental spiritual, sehat dan berbudaya.

2. Misi
   a. Melaksanakan pengembangna kurikulum SMP Negeri 28 Bandung.
   b. Melaksanakan pengembangan standar pengelolaan sekolah.
   c. Melaksanakan peningkatan kompetensi pendidik dan tenga kependidikan.
   d. Melaksanakan pengembangan sarana dan prasarana.
   e. Mempertahankan stabilitas sekolah.
   f. Melaksanakan pengembangna kehidupan bernuansa religi.
   g. Melaksanakan pengembangan budaya nasional.
   h. Melaksanakan pengembangan perilaku hidup bersih dan sehat.
   i. Melaksanakan pengembangna sekolah peduli lingkungan hidup.
   j. Melaksanakan kegiatan pengolahan sampah melalui 3R.
   k. Melaksanakan kegiatan penghijauan lingkungan sekolah.
   l. Menggiatkan pembuatan biopori.

II.1.4 Struktur Organisasi



Gambar II.2 Struktur Organisasi

II.1.5 Deskripsi Tugas dan Tanggung Jawab

Deskripsi tugas dan tanggung jawab digunakan untuk mengetahui tugas, wewenang dan tanggung jawab dari masing-masing bagian. Adapun deskripsi tugas secara garis besar yang ada di SMP Negeri 25 Bandung adalah sebagai berikut:

1. Kepala sekolah
   a. Bertugas sebagai edukator
   b. Manajer administrator
   c. *Supervisor*
   d. Pemimpin
   e. Inovator
   f. Motivator
2. Kepala Tata Usaha
   a. Menyusun program tata usaha sekolah
   b. Mengelola administrasi keuangan sekolah
   c. Mengelola administrasi kesiswaan
   d. Mengelola administrasi perlengkapan
   e. Mnegelola statistika sekolah
   f. Memberi layanan administrais kepda guru dan siswa

g. Menata dan melaksanakan pemeliharaan sekolah
h. Memfasilitasi guru dalam pelaksanaan tugas atas izin sekolah
i. Menyusun laporan

3. Wakasek Kesiswaan
   a. Menyusun program pembinaan kesiswaan
   b. Menyusun sistem masa orientasi siswa (MOS)
   c. Menyusun sistem diteksi pelanggaran disiplin siswa
   d. Mengkoordinasi Pembinaan kesiswaan
   e. Mengkoordinasikan data kehadiran siswa
   f. Mengatur perijinan siswa untuk melaksanakan kegiatan diluar sekolah

   Menyusun laporan kegitan kesiswaan
4. Wakasek Kurikulum
   a. Menyusun program pengjaran
   b. Menyusun dan memiliki sistem informasi kurikulum untuk semua guru
   c. Menyusun tugas guru dan jadwal piket guru
   d. Menyusun kriteria indikator pencapaian program kenaikan dan kelulusan
   e. Menyusun laporan kegiatan akademis
   f. Mengembangkan sistem evaluasi
   g. Membina lomba bidang akademis
5. Wakasek Hubungan Masyarakat
   a. Melakukan perencanaan dan program kerja sama dengan masyarakat
   b. Mengembangkan konsep anggaran ruamh tangga untuk melancarakan kerja sama dengan komite sekolah
   c. Memfasilitasi hubungan antar sekolah
   d. Mengembangkan peluang kerja sama antara siswa dan guru sert sumber daya lingkungan masyarakat
   e. Mengkoordinasi pertemuan orang tua siswa
   f. Mengatur pernyusunan dan penyimpanan agenda rapat
   g. Menyusun laporan pelaksanaan program hubungan dengan masyarakat
6. Wakasek Sarana dan Prasarana
   a. Mengembangkan desain penataan lingkungan sekolah

b. Melakukan penataan dan pemeliharaan lingkungan bersih sekolah
c. Mengembangkan sekolah sebagai ekosistem yang sehat serta edukatif
d. Mengatur jadwal piket untuk menjaga kebersihan sekolah
e. Mengkoordinasikan pembangunan dan pemeliharaan bangunan
f. Memfasilitasi penyediaan sarana guru dan siswa
g. Menyusun program pemeliharaan dan pemberdayaan sarana kantor dan belajar
h. Membantu guru-guru dalam mengembangkan media belajar
i. Menyusun laporan pelaksanaan sarana dan prasarana

7. Guru/Wali Kelas
   a. Melakukan kegiatan mengajar
   b. Pembimbing siswa
   c. Pembina siswa
   d. Konsultan siswa
   e. Pengarah berbagai kegiatan siswa
   f. Mengumpulkan dan menghitung angka kredit/kenaikan pangkatnya
   g. Memeriksa, menilai dan mengumumkan hasil tugas siswan
   h. Mengisi daftar siswa
8. Siswa

   Mengikuti aturan sekolah sebagai pelajar

II.2 Landasan Teori

Dasar-dasar dalam pembangunan perangkat lunak sebagai bahan acuan. Berisi teori-teori yang bisa dijadikan dasar dan acuan dalam perancangan perangkat lunak.

II.2.1 Sekolah

Sekolah adalah tempat peserta didik untuk dapat memperoleh pendidikan. Sekolah dapat diartikan sebagai sebuah lembaga tempat proses belajar-mengajar pada sebuah sistem pendidikan yang diakui oleh negara. Meskipun demikian terdapat juga beberapa sistem pendidikan yang bertujuan layaknya seperti sekolah formal dengan bentuk yang berbeda seperti yang dikenal istilah home schooling,

akan tetapi sekolah adalah sistem pendidikan yang paling terkenal bahkan ada di setiap negara [2].

Menurut Kamus besar bahasa Indonesia sekolah adalah bangunan atau lembaga belajar dan mengajar serta tempat menerima dan memberi pelajaran menurut dan sesuai tingkatan yang ditempuh. Tingkatan ini selanjutnya disebut jenjang yang ada, yakni tingkatan adalah sekolah dasar, Lanjutan dan Perguruan tinggi [2].

II.2.2 Data Survey

Data survei yang digunakan pada penelitian ini menggunakan metode wawancara. Wawancara dilakukan dengan narasumber Wakil Kepala Sekolah Bagian Kesiswaan dan Petugas Perpustakaan.

II.2.3 Konsep Rekayasa Perangkat Lunak

Perangkat lunak merupakan program-program komputer dan dokumentasi yang berkaitan dengan program tersebut. Sedangkan definisi dari rekayasa perangkat lunak sendiri yaitu disiplin ilmu rekayasa atau teknik yang berkaitan dengan semua aspek dalam membuat perangkat lunak. Perangkat lunak memiliki beberapa karakteristik diantaranya [3]:

1. *Maintanability* (dapat dirawat) yaitu perangkat lunak harus dapat memenuhi perubahan kebutuhan.
2. *Dependability* yaitu perangkat lunak harus dapat dipercaya.
3. *Efisiensi* yaitu perangkat lunak harus efisien dalam penggunaan *resource*.
4. *Usability* yaitu perangkat lunak harus digunakan sesuai dengan apa yang direncanakan.

II.2.4 Android

Android merupakan sistem operasi *mobile* yang tumbuh di tengah sistem operasi lainnya yang berkembang dewasa ini. Sistem operasi lainnya seperti *Windows Mobile, i-Phone OS, Symbian* dan masih banyak lagi juga menawarkan

kekayaan isi dan keoptimalan berjalan di atas perangkat hardware yang ada. Akan tetapi, sistem operasi yang ada ini berjalan dengan memprioritaskan aplikasi inti yang dibangun sendiri tanpa melihat potensi yang cukup besar dari aplikasi pihak ketiga. Oleh karena itu, adanya keterbatasan distribusi aplikasi pihak ketiga untuk *platform* mereka [4].

II.2.4.1 Android *Life Cycle*

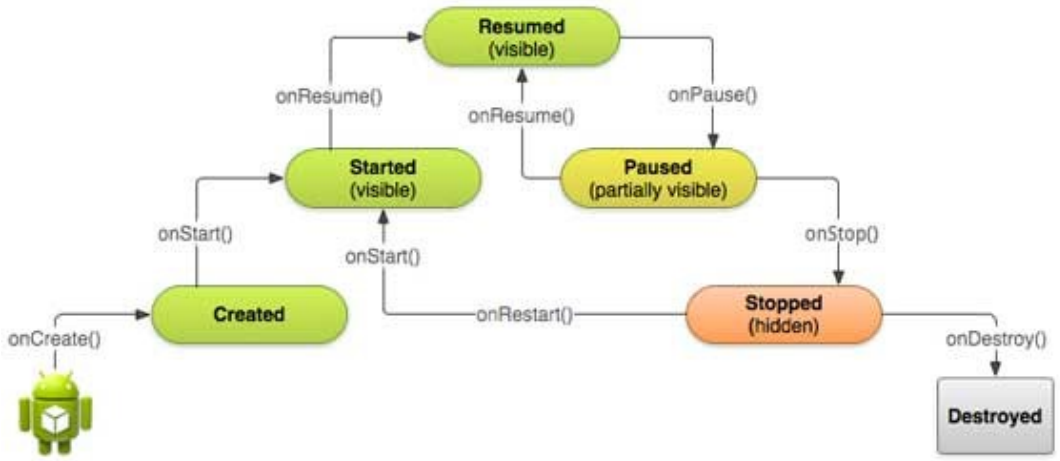
Apikasi Android terdiri dari beberapa fungsi dasar seperti mengedit catatan, memutar file musik, membunyikan *alarm*, atau membuka kontak telepon. Fungsi-fungsi tersebut dapat diklasifikasikan ke dalam empat komponen Android yang berbeda seperti ditunjukkan pada tabel II.1, klasifikasi tersebut berdasarkan kelas-kelas dasar *Java* yang digunakan [4].

Tabel II.1 Komponen Aplikasi Android

Functionality	Java Base Class	Example
Focused thing a user can do	Activity	Edit a note, play a game
Background process	Service	Play music, update weather icon
Receive messages	BroadcastReceiver	Trigger alarm upon event
Store and retrive data	ContentProvider	Open a phone contact

Setiap aplikasi pasti menggunakan minimal satu dari komponen tersebut, akan tetapi terdapat beberapa komponen yang mengharuskan mencantumkan *specified permission* sebelum digunakan seperti komponen *Service, BroadcastReceiver, ContentProvider* [4].

Android memiliki paradigma pemrograman lain tidak seperti paradigma pemrograman biasa dimana aplikasi yang dijalankan pada fungsi main(), sistem Android menjalankan kode dalam *method Activity* dengan menerapkan metode *callback* tertentu yang sesuai dengan tahap tertentu dari siklus hidup. Setiap aplikasi yang berjalan dalam sistem operasi Android memiliki siklus hidup yang berbeda dengan aplikasi *dekstop* atau web. Hal ini dikarenakan aplikasi *mobile* memiliki tingkat *interupsi* proses yang lumayan tinggi seperti ketika *handling* panggilan masuk aplikasi diharuskan menghentikan proses sementara. Penerapan siklus hidup juga berguna untuk memastikan aplikasi tidak menghabiskan sumber daya baterai pengguna [5].



Gambar II.3 Siklus Hidup Android [5]

II.2.5 NFC

Near Field Cmmunication (NFC) adalah satu set teknologi nirkabel jarak pendek yang membutuhkan jarak kurang lebih 4cm. NFC digunakan untuk berbagi data melalui tag NFC dan perangkat android.

Tag NFC memiliki kompleksitas yang beragam. Tag sederhana hanya dapat membaca dan menulis tag, yang terkadang hanya digunakan satu kali dan di jadikan *read-only.* Tag yang lebih kompleks bisa digunakan dengan operasi matematika dan memiliki kriptografi yang digunakan untuk autentikasi pada pembacaan tag NFC. Tag yang paling canggih.

Near Field Communication (NFC) merupakan teknologi komunikasi baru dengan menggunakan induksi magnet berbasis teknologi *Radio Frequency Identification* (RFID). NFC mulai dikembangkan pada tahun 2002 oleh NXP dan Sony. Mereka berhasil membangun Forum NFC pertama bersama dengan Nokia. Sejak saat itu Forum NFC memiliki lebih dari 150 anggota dan peneliti yang terlibat dari beberapa perusahaan dan organisasi. Pada forum ini juga NFC dipromosikan dan secara resmi mengeluarkan sertifikasi untuk perangkatperangkat yang mengadopsi teknologi NFC. NFC bisa juga disebut sebagai generasi kedua dari teknologi RFID.

NFC beroperasi pada frekuensi 13,56 MHz dengan kecepatan transmisi pengiriman mencapai 424 kbit/s. Jarak transmisi NFC sekitar 4-10 cm. Perbedaan antara NFC dan teknologi komunikasi *contactless* lainnya yaitu perangkat NFC dapat bersifat aktif – aktif (*peer to peer*) dan aktif – pasif. Oleh karena itu NFC selalu melibatkan inisiator (*reader*) dan target. Inisiator aktif menghasilkan medan RF (*Radio Frequency*) yang dapat memberikan kekuatan ke target yang pasif (tidak memiliki sumber daya). Hal ini memungkinkan target NFC untuk memiliki bentuk yang sangat sederhana seperti stiker, gantungan kunci, atau kartu yang tidak memerlukan energy khusus [6].

II.2.5.1 NFC *Tag*

NFC *Tag* merupakan media pendukung teknologi NFC. NFC Tag biasanya bersifat pasif, sehingga membutuhkan inisiator (NFC bersifat aktif) untuk proses komunikasi. NFC *Tag* dapat berisi data dengan ukuran kecil. NFC *Tag* saat ini memiliki memory berkisar antara 96 dan 4.096 byte. NFC *Tag* berisfat *read-only* tetapi memungkinkan juga bersifat *re-writeable* [7].



Gambar II.4 NFC Tag [7]

Produsen NFC telah menentukan standar dari NFC *Tag* untuk dapat dioperasikan dengan perangkat NFC. Spesifikasi yang ditentukan terbagi menjadi 4 tipe. Keempat tipe *Tag* NFC tersebut didasarkan pada produk dan ketersediaan komersil. Berikut adalah 4 tipe dari NFC *Tag*:

1. Tipe 1 : didasari pada ISO/IEC 14443A. *Tags* mampu membaca dan menulis ulang. Pengguna dapat mengatur tag menjadi *read-only*. Memori yang tersedia adalah 96 bytes dan dapat di-*upgrade* hingga 2 Kbyte.

2. Tipe 2 : didasari pada ISO/IEC 14443A. *Tags* mampu membaca dan menulis ulang. Pengguna dapat mengatur tag menjadi *read-only*. Memori yang tersedia adalah 48 bytes dan dapat di upgrade hingga 2 Kbyte.
3. Tipe 3 : didasari pada standar industri Jepang atau *Japanese industrial standard* (JIS) X 6319-4, atau disebut Felica. *Tag* diatur pada pembuatannya untuk mendukung baik dalam *read* dan *re-writeable*, ataupun *read-only*. Memori yang tersedia dapat diubah, secara teori batas memorinya adalah 1 Mbyte per *service*.
4. Tipe 4 : kompatibel sepenuhnya dengan ISO/IEC 14443 standard *series*, memiliki memori 32 KB dan lebih cepat daripada tipe *tag* lainnya, memiliki proteksi terhadap data *collision*, dapat menggunakan komunikasi NFC-A maupun NFC-B. Tipe 4 dapat ditulis ulang *(rewriteabl*e) atau *read-only*. Tetapi pengaturan tersebut dilakukan saat tipe 4 dalam tahap pembuatan. Saat sudah selesai dibuat, pengaturannya tidak dapat diubah [7].

II.2.5.2 Perangkat Android Dengan NFC

Perangkat android mendukung tiga mode operasi, yaitu [8]:

1. *Reader/Writer Mode,* memungkinkan perangkat NFC untuk membaca dan menulis tag NFC pasif dan stiker NFC.
2. *P2P Mode,* memungkinkan perangkat NFC bertukar data dengan perangkat NFC lainnya. Mode ini digunakan oleh *Android Beam.*
3. *Card Emulation Mode,* memungkinkan perangkat NFC sebagai kartu NFC sendiri, kartu NFC ini dapat diakses oleh pembaca NFC eksternal, seperti *NFC point-of sale.*

II.2.5.3 NFC *Basic*

NFC Basic menjelaskan tentang tugas dasar yang dilakukan android. *NFC Basic* menjelaskan bagaimana cara mengirim dan menerima data NFC dalam bentuk *NDEF Messages* dan menggambarkan *framework API* yang mendukung fitur ini [8].

Ada dua fungsi yang digunakan ketika bekerja dengan *NDEF data* dan Android [8]:

1. Membaca data NDEF dari tag NFC

   Membaca data NDEF dati tag NFC ditangani oleh *tag dispatch system* yang ada pada tag NFC dan mengkategorikannya dengan tepat. *Tag Dispatch System* merupakan sebuah aplikasi yang menangani pemindaian tag NFC untuk memindai dan mendeklarasikan permintaan yang dimaksud dan untuk menangani data yang dikirim.
2. *Beaming NDEF Message* dari satu perangkat ke perangkat lain dengan *Android Beaming.*

   Fitur *Android Beaming* memungkinkan perangkat android untuk mengirim *NDEF messages* ke perangkat lain dengan menekan perangkat android secara bersamaan. Interaksi ini lebih mudah dibandingkan dengan teknologi nirkabel lain seperti *Bluetooth,* karena dengan NFC, sambungan ini otomatis akan dimulai ketika dua perangkat saling berdekatan. *Android Beam* tersedia dalam NFC API, sehingga antar perangkat bisa saling mengirim informasi.

II.2.5.4 *Tag Dispatch System*

Perangkat android akan mencari tag NFC saat layar terkunci, terkecuali saat NFC dinonaktifkan. Ketika android menemukan tag NFC, harapan yang diinginkan adalah sebuah perangkat android memiliki aktivitas yang paling tepat untuk menangani *intent* tanpa meminta pengguna untuk memilih aplikasi apa yang akan digunakan untuk menanganinya. Untuk menangani hal tersebut, android telah menyediakan sistem pengiriman tag NFC khusus yang dapat menganalisa pemindaian tag NFC, memparsing data NFC, dan menemukan aplikasi yang tepat dengan data yang ada pada NFC. Hal ini dilakuakn dengan [8]:

1. Menguraikan tag NFC dan mencari tahu jenis MIME atau URI yang berfungsi untuk mengenali data *payload* yang terdapat dalam tag.
2. Enkapsulasi jenis MIME atau URI dan *payload* kedalam intent.
3. Memulai *activity* berdasarkan intent.

II.2.5.5 Pemetaan Tag NFC

Ketika sebuah perangkat android memindai tag NFC yang berisi data NDEF, maka perangkat android akan mengurai pesan dan mencari tahu tipe data

apakah MIME atau URI. Untuk melakukan hal ini, pertama sistem membaca NDEF *record* didalam NDEF *message* untuk menterjemahkan seluruh pesan NDEF. Dalam NDEF *message,* NDEF *record* pertama berisi kolom berikut [8]:

1. 3-bit TNT (*Type Name Format*)

   Menunjukan bagaimana menterjemahkan panjang jenis variable. Nilai yang valid dijelaskan pada table II.2.

2. *Variable Length Type*

   Menjelaskan jenis *record.* Jika menggunakan TFT_WELL_KNOWN, gunakan bidang ini untuk menentukan *Record Type Definition* (RTD). Nilai RTD dijelaskan pada table II.3.

3. *Variable Length ID*

   Sebuah pengenal unik untuk mengenali *record.* Bidang ini tidak terlalu sering digunakan, tapi dibutuhkan jika akan membuat id pada sebuah tag.

4. *Variable Length Payload*

   Payload yang digunakan untuk membaca dan menulis tag NFC. NDEF *messages* dapat berisi beberapa NDEF *record.*

Sistem pengiriman tag NFC menggunakan TNF (*Type Name Format*) dan jenis kolom untuk memetakan jenis MIME atau URI untuk NDEF. Jika berhasil, sistem akan menyatukan informasi kedalam ACTION_NDEF_DISCOVERY intent bersama dengan payload yang sebenarnya. Namun, terdapat masalah seperti *tag dispatch system* tidak dapat menemukan jenis data berdasarkan NDEF *record.* Hal ini terjadi ketika data NFC tidak bisa dipetakan kedalam jenis MIME atau URI. Atau ketika tag NFC tidak berisi data NDEF. Dalam kasus tersebut, objek tag yang memiliki informasi tentang teknologi tag dan payload yang dikemas dalam sebuah ACTION_TECH_DISCOVERY [8].

Table II.2 menjelaskan bagaimana *tag dispatch system* dipetakan dalam TNF dan tipe kolom untuk jenis MIME atau URI, juga menjelaskan TNF yang tidak dapat dipetakan kedalam jenis MIME atau URI, dalam kasus ini *tag dispatch system* akan memanggil kembali ACTION_TECH_DISCOVERY [8].

Tabel II.2 Pemetaan TNF

***Type Name Format* (TNF)**	**Pemetaan**
TNF_ABSOLUTE_URI	Jenis kolom URI
TNF_EMPTY	Kembali ke ACTION_TECH_DISCOVERY
TNF_EXTERNAL_TYPE	Jenis kolom URI berdasarkan URN. Dikodekan kedalam jenis NDEF dalam bentuk: *<domain-name>:<service-name>*. Android memetakan ini kedalam URI dalam bentuk: *vnd.android.nfc://ext/<domain_name>:<service-name>*.
TNF_MIME_MEDIA	Jenis kolom MIME
TNF_UNCHANGED	*Record* pertama tidak valid, kembali ke ACTION_TECH_DISCOVERY
TNF_UNKNOW	Kembali ke ACTION_TECH_DISCOVERY
TNF_WELL_KNOWN	Tipe MIME atau URI tergantung *Record Type Definition* (RTD).

Tabel II.3 Pemetaan RTD untuk TNF_WELL_KNOWN

Record Type Definition (RTD)	**Pemetaan**
RTD_ALTERNATIVE_CARRIER	Kembali ke ACTION_TECH_DISCOVERY
RTD_HANDOVER_CARRIER	Kembali ke ACTION_TECH_DISCOVERY
RTD_HANDOVER_REQUEST	Kembali ke ACTION_TECH_DISCOVERY
Record Type Definition (RTD)	**Pemetaan**
RTD_HANDOVER_SELECT	Kembali ke ACTION_TECH_DISCOVERY
RTD_SMART_POSTER	URI berdasarkan parsing payload
RTD_TEXT	Jenis MIME *text/plain*
RTD_URI	URI berdasarkan payload

II.2.5.6 Pengiriman Tag NFC ke Aplikasi

Ketika *tag dispact system* berhasil mengirimkan intent yang telah dikemas kedalam tag NFC, sistem akan mengirimkan intent sesuai dengan data pada tag NFC. Jika lebih dari satu aplikasi bisa menangani intent tersebut, maka pemilihan *activiry* akan ditampilkan, sehingga pengguna dapat memilih *activity.* Tag dispatch system menentukan tiga intent. Berikut daftar intent dalam urutan tertinggi ke terendah [8]:

1. ACTION_NDEF_DISCOVERED

   Intent ini digunakan untuk memulai *activity* ketika tag yang berisi payload NDEF dipindai dan dari jenis yang dikenal. Ini adalah intent dengan prioritas tertinggi, dan *tag dispatch system* akan mencoba memulai sebuah *activity* dengan intent ini sebelum intent lain.

2. ACTION_TECH_DISCOVERED

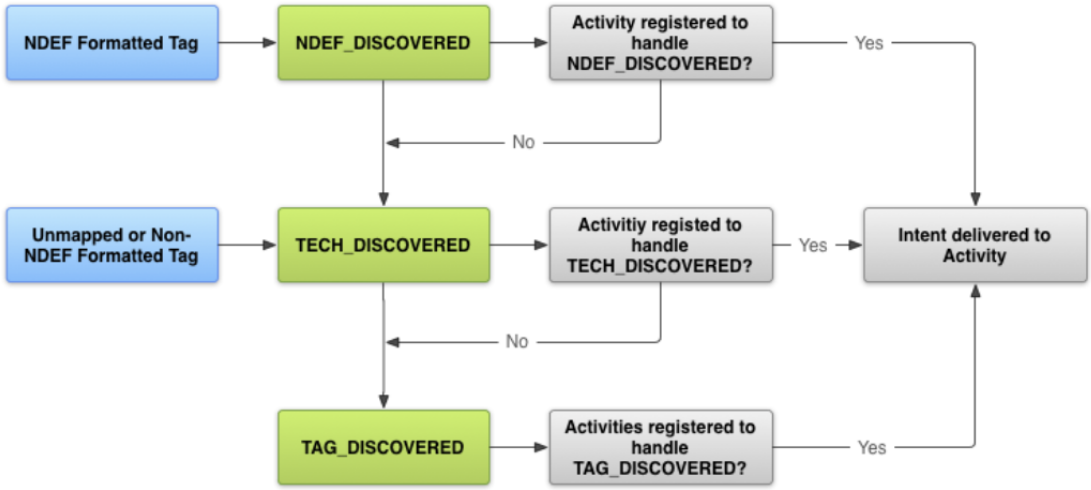
   Jika tidak ada *activities* yang dapat menangani intent ACTION_NDEF_DISCOVERED, *tag dispatch system* akan mencoba memulai aplikasi yang sesuai dengan intent. Intent ini langsung dimuali jika tag yang dipindai berisi data NDEF yang tidak dapat dipetakan kedalam jenis MIME atau URI. Atau jika tag tidak memiliki data NDEF tetapi dari teknologi yang dikenal.

3. ACTION_TAG_DISCOVERED

   Intent ini dimulai jika tidak ada *activity* yang dapat menangani ACTION_NDEF_DISCOVERED atau ACTION_TECH_DISCOVERED.

Cara dasar *tag dispatch system* bekerja adalah sebagai berikut [8]:

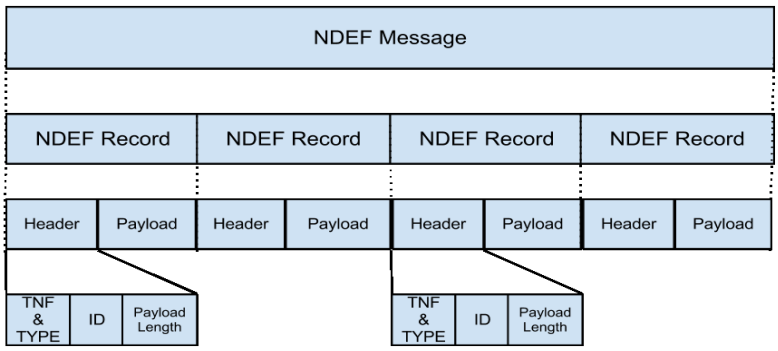
1. Coba memulai *activity* dengan intent yang telah dibuat oleh *tag dispatch system* ketika mengurai tag NFC (antara ACTION_TECH_DISCOVERED atau ACTION_TECH_DISCOVERED).
2. Jika tidak ada aktivitas untuk menangani intent, coba untuk memulai *activity* dengan prioritas rendah (antara ACTION_TECH_DISCOVERED atau ACTION_TECH_DISCOVERED) sampai aplikasi menemukan intent atau sampai *tag dispatch system* mencoba semua intent yang memungkinkan.
3. Jika tidak ada aplikasi yang sesuai maka sistem tidak akan melakukan apapun.



Gambar II.5 Tag Dispatch System [8]

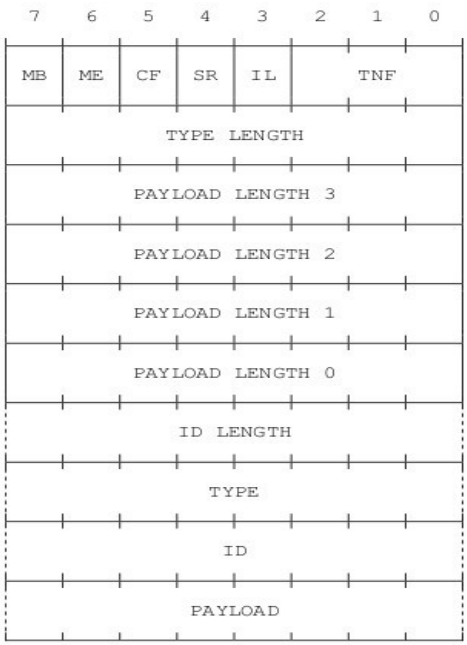
II.2.5.7 NFC *Data Exchange Format* (NDEF)

NFC Data Exchange Format (NDEF) adalah format pesan standar pada proses pertukaran informasi antara perangkat NFC ke perangkat NFC dan perangkat NFC ke *Tag* NFC. Di dalam NDEF terdapat NDEF *Message* yang terdiri dari beberapa *record* seperti pada gambar 2.21 [10].



Gambar II.6 Struktur NDEF Message [7]

NDEF *Record* merupakan format umum dari panjang variabel *record.* Berikut adalah struktur NDEF *Record.*



Gambar II.7 Struktur NDEF Record [7]

Berikut adalah keterangan dari struktur NDEF *Record*:

1. MB (*Message Begin*) : Area 1 bit. Ketika diset maka akan menunjukan awal NDEF *Message*.
2. ME (*Message End*) : Area 1 bit. Ketika diset maka akan menunjukan akhir NDEF *Message*.
3. CF (*Chunk Flag*) : Area 1 bit. Untuk mengidentifikasi apakah potongan *record* pertama atau potongan tengah *record*.
4. SR (Short Record) : Area 1 bit. Jika diset maka PAYOAD_LENGTH merupaan oktet tunggal. Short Record ini dimaksudakn untuk mengkapsulasi *Payloadi,* dengan kisaran ukuran 0 sampai 255
5. IL (*Identification Length*) : Area 1 bit. Jika diset maka ID_LENGTH akan muncul di bagian *header* sebagai oktet tunggal. Jika ID_LENGTH bernilai 0. ID_LENGTH

dihilangkan dari *header record* dan area ID dihilangkan dari *record.*

6. TNF (*Type Name Format*) : Area bernilai 3 bit. Untuk mengidentifikasi struktur dari *value* TYPE.

Tabel II.4 Value Type Name Format

Type Name Format	***Value***
Empty	0x00
NFC Forum well-known type [NFC RTD]	0x01
Media-type as defined in RFC 2046 [RFC 2046]	0x02
Absolute URI as defined in RFC 3986 [RFC 3986]	0x03
NFC Forum external type [NFC RTD]	0x04
Unkown	0x05
Unchanged	0x06
Reserved	0x07

Keterangan Tabel II.4 Value Type Name Format:

1. 0x00 (Empty)

   Menunjukan bahwa tidak ada jenis atau payload yang terkait dengan *record*. Ketika digunakan TYPE_LENGTH, ID_LENGTH dan bidang PAYLOAD_LENGTH harus bernilai nol dan TYPE_ID dan PAYLOAD dihilangkan dari *record*. Nilai TNF ini dapat digunakan setiap kali *record* kosong diperlukan, misalnya untuk mengakhiri NDEF *message* ketika tidak ada PAYLOAD yang diinputkan oleh pengguna aplikasi.
2. 0x01 (NFC Forum wel-known type [NFC RTD])

   Menunjukan bahwa TYPE berisi nilai yang mengikuti *Type Name Format* RTD sesuai spesifikasi pada NFC Forum.
3. 0x02 (Absolute URI as defined in RFC 3986 [RFC 3986])

   Menunjukan bahwa TYPE berisi nilai yang sesuai dengan yang dibuat oleh BNF dan didefinisikan dalam RFC 2046.
4. 0x03 (Absolute URI as defined in RFC 3986 [RFC 3986])

Menunjukan bahwa TYPE berisi nilai yang mutlak sesuai dengan URI BNF dan didefinisikan dalam RFC 3986.

5. 0x04 (NFC Forum external type [NFC RTD])
   Menunjukkan bahwa bidang TYPE berisi nilai yang mengikuti *Type Name Format* RTD yang didefinisikan dalam NFC Forum RTD spesifikasi [NFC RTD] untuk nama jenis eksternal.
6. 0x05 (Unkown)
   Menunjukan jika tipe PAYLOAD tidak diketahui. Hal ini serupa dengan "application/octet-stream" jenis media yang didefinisikan oleh MIME RFC 2046. Ketika digunakan TYPE_LENGTH harus bernilai nol. Maka dengan demikian TYPE dihilangkan dari NDEF *record*.
7. 0x06 (Unchanged)
   Hanya untuk *record* tengah dan mengakhiri *record* dari Chunked Payload. TYPE_LENGTH harus bernilai nol.
8. 0x07 (Reserved)
   Milik NFC Forum untuk digunakan di masa depan.
9. TYPE_LENGTH
   TYPE_LENGTH adalah 8bit integer yang menentukan panjang oktet dari TYPE
10. ID_LENGTH
    ID_LENGTH adalah 8bit integer yang menentukan panjang oktet dalam ID. ID_LENGTH muncul apabilan IL diset menjadi 1 di dalam *record header*. ID_LENGTH boleh bernilai 0 oktet namun ID harus dihilangkan dari NDEF *record*.
11. PAYLOAD_LENGTH
    PAYLOAD_LENGTH adalah yang menentukan panjang oktet dari PAYLOAD.
12. TYPE
    Nilai TYPE adalah identifikator yang menjelaskan jenis nilai PAYLOAD. Nilai dari TYPE harus sesuai dengan struktur, encoding dan format pada TNF.
13. ID

Nilai dari ID adalah identifier dalam bentuk resferensi URI yang dijelaskan dalam RFC 3986. Referensi URI dapat bersifat relative dan absolut karena NDEF tidak mendefiniskan basis URI.

14. PAYLOAD

    Konten yang dapat berisi apa saja dalam ukuran beberapa bytes.

II.2.6 GCM (*Google Cloud Messaging*)

GCM (*Google Cloud Messaging*) adalah servis gratis yang disediakan oleh google yang memungkinkan para pengembang untuk mengirim pesan pada aplikasi *client-server*. Pengiriman pesan ini bisa dilakukan dari server ke client ataupun dari client ke server.

II.2.6.1 *Lifecycle Flow*

1. Mendaftar untuk medapatkan akses GCM [9].
   Aplikasi client mendaftar untuk menerima pesan
2. Mengirim dan menerima pesan *downstream* [9].
   a. Mengirim pesan. Aplikasi server mengirim pesan kepada aplikasi client:
      i. Aplikasi server mengirim pesan ke GCM *Connection Server.*
      ii. GCM *Connection Server* mengurutan dan menyimpan pesan jika perangkat dalam keadaan mati.
      iii. Jika perangkat dalam keadaan hidup, GCM *Connection Server* akan mengirimkan pesan ke perangkat tersebut.
   b. Menerima pesan. Aplikasi client menerima pesan dari GCM *Connection Server*
3. Mengirim dan menerima pesan *upstream* [9].
   Fitur ini hanya tersedia jika menggunakan protocol XMPP.
   a. Mengirim pesan, aplikasi client mengirim pesan ke aplikasi server.
      i. Aplikasi server mengirim pesan ke XMPP *Connection Server*
      ii. XMPP *Connection Server* mengurutkan dan menyimpan pesan jika perangkat dalam keadaan mati.

iii. Ketika perangkat dalm keadaan hidup, XMPP *COnnction Server* akan mengirimkan pesan ke aplikasi server.

b. Menerima pesan, aplikasi server menerim pesan dari XMPP *Connection Server* dan kemudian melakukan hal berikut:
   i. Mengurai pesan untuk memverifikasi informasi pengiriman pesan dari aplikasi client.
   ii. Mengirimakan "*ack*" ke XMPP *Connectio Server* untuk mengkonfirmasi penerimaan pesan.
   iii. Mengurai payload pesan, seperti yang didefinisikan oleh aplikasi client.

II.2.7 Web Service

Web service Sebuah layanan web adalah standar yang digunakan untuk bertukar informasi antara aplikasi atau sistem jenis *heterogen*. Aplikasi perangkat lunak yang ditulis dalam berbagai bahasa pemrograman dan berjalan pada berbagai platform dapat menggunakan layanan *web* untuk bertukar informasi melalui Internet dengan menggunakan protokol http. Interoperabilitas ini dapat dicapai antara aplikasi Java dan Dot net, atau PHP dan aplikasi Java [10].

Beberapa keuntungan menggunakan *web service* pada Android yaitu [11]:

1. Dengan menambahkan *web service* akan membuat perangkat pengguna bekerja menjadi lebih ringan, baik dari segi daya CPU yang diperlukan dan bandwidth yang digunakan.
2. *Web service* juga bisa dimanfaatkan dengan signifikan dengan menggunakan kembali fungsi yang ada sebagai metode web.
3. *Database* bisa di akses melalui aplikasi Android dari jarak jauh.

II.2.8 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 Pemrograman JavaScript, Standar ECMA-262 Edisi ke 3 – Desember 1999. JSON

merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data. JSON terbuat dari dua struktur:

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), daftar berkunci (*keyed list*), atau *associative* array [11].
2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*) [11].

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

II.2.9 API

API adalah sekumpulan perintah, fungsi, dan protokol yang dapat digunakan saat membangun perangkat lunak untuk sistem operasi tertentu. API memungkinkan programmer untuk menggunakan fungsi standar untuk berinteraksi dengan system operasi. API atau *Application Programming Interface* juga merupakan suatu dokumentasi yang terdiri dari antar muka, fungsi, kelas, struktur untuk membangun sebuah perangkat lunak.

Dengan adanya API, maka memudahkan seorang programmer untuk membongkar suatu *software* untuk kemudian dapat dikembangkan atau diintegrasikan dengan perangkat lunak yang lain. API dapat dikatakan sebagai penghubung suatu aplikasi dengan aplikasi lainnya. Suatu rutin standar yang memungkinkan *developer* menggunakan *system function*. Proses ini dikelola

melalui *operating system*. Keunggulan dari API ini adalah memungkinkan suatu aplikasi dengan aplikasi lainnya untuk saling berinteraksi [12].

Keuntungan dengan menggunakan API adalah sebagai berikut:

1. Portabilitas.

   Developer yang menggunakan API dapat menjalankan programnya dalam sistem operasi mana saja asalkan sudah terinstal API tersebut.
2. Lebih Mudah Dimengerti

   API menggunakan bahasa yang lebih terstruktur dan mudah dimengerti daripada bahasa *system call*. Hal ini sangat penting dalam hal *editing* dan pengembangan.

Cara menggunakan API:

1. Dilakukan dengan mengimpor package/kelas
2. Ada beberapa kelas bernama sama dipackage yang berbeda, yaitu:
   a. Import salah satu dan gunakan nama lengkap untuk yang lain
   b. Gunakan nama lengkap semua kelas

Kebanyakan Sistem Operasi seperti Windows, menyediakan fasilitas API sehingga programmer dapat melakukan aktivitas programming dengan lebih konsisten. Meskipun API didesain untuk programer, namun API juga baik untuk user karena setidaknya dapat menjamin bahwa program tersebut memiliki *interface* yang sama, sehingga lebih mudah untuk dipelajari.

II.2.10 *Object Oriented Analysis Desain*

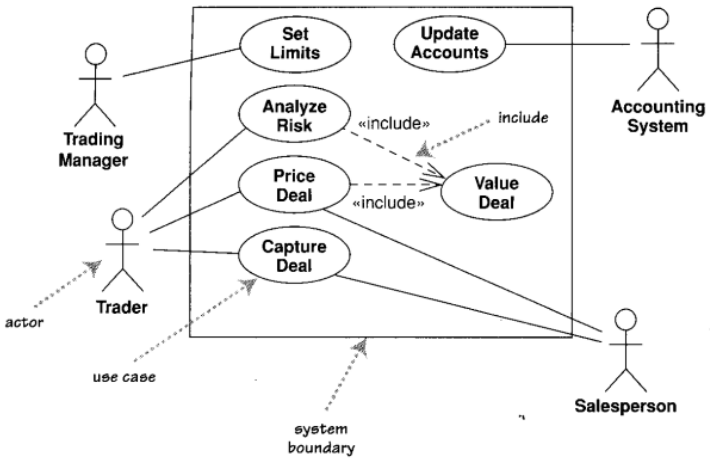
Konsep OOAD mencakup analisis dan desain sebuah sistem dengan pendekatan objek, yaitu analisis berorientasi objek (OOA) dan desain berorientasi objek (OOD). OOA adalah metode analisis yang memeriksa requirement (syarat/keperluan) yang harus dipenuhi sebuah sistem dari sudut pandang kelas-kelas dan objek-objek yang ditemui dalam ruang lingkup sistem. Sedangkan OOD adalah metode untuk mengarahkan arsitektur software yang didasarkan pada manipulasi objek-objek sistem atau subsistem [13].

II.2.11 *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah termasuk ke dalam rumpun jenis pemodelan notasi grafis yang didukung oleh model-model tunggal. Pemodelan ini berguna untuk membantu dalam menjelaskan data rancang perangkat lunak yang dibangun dengan *object-oriented* (OO). UML merupakan standar terbuka yang dikelola *Open Management Group* (OMG) yang berada dibawah naungan perusahaan-perusahaan konsorium terbuka. UML merupakan suatu bahasa pemodelan yang terdiri banyak model diantaranya adalah [13]:

II.2.10.1 *Use Case Diagram*

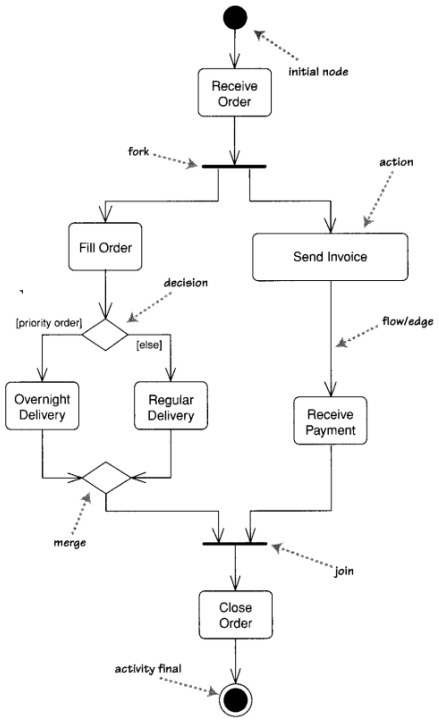
Use Case Diagram merupakan pemodelan untuk menggambarkan kelakuan (*behavior*) system secara keseluran yang akan dibuat. Diagram *use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Dengan pengertian yang cepat, diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Yang ditekankan pada diagram ini adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". *Use case* menjelaskan secara sederhana fungsi sistem dari sudut pandang user [13]:



Gambar II.8 Use Case Diagram [13]

II.2.10.2 *Activity Diagram*

Activity Diagram adalah sebuah tahapan yang lebih fokus kepada menggambarkan proses bisnis dan urutan aktivitas dalam sebuah proses. Dimana biasanya dipakai pada business modeling untuk memperlihatkan urutan aktifitas proses bisnis. *Activity Diagram* ini sendiri memiliki struktur diagram yang mirip *Flowchart* atau *data flow diagram* pada perancangan terstruktur. *Activity Diagram* dibuat berdasarkan sebuah atau beberapa *use case* pada *Use Case Diagram* [13].

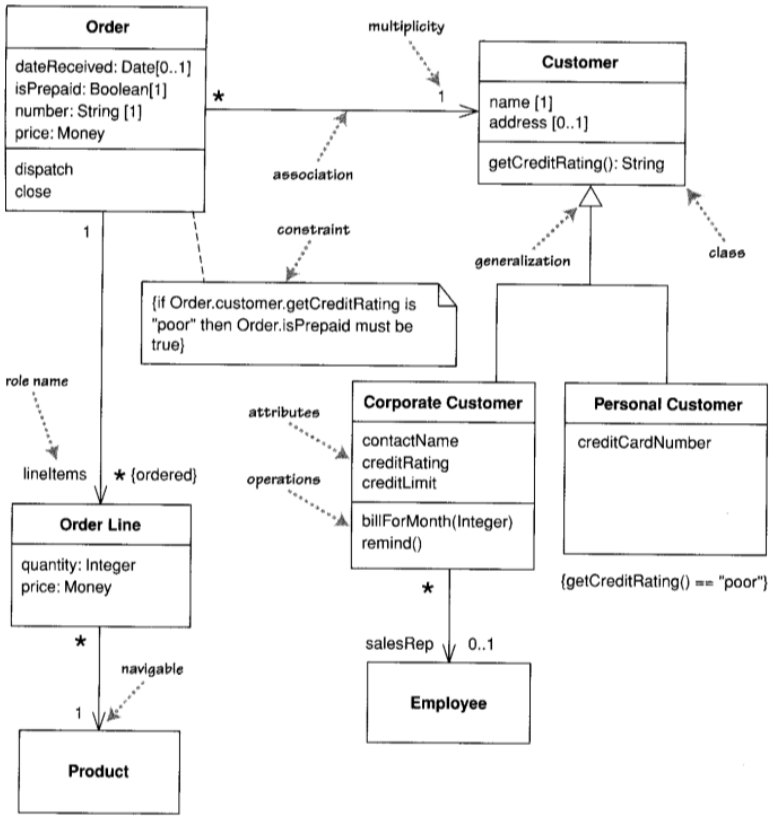


Gambar II.9 Activity Diagram [13]

II.2.10.3 *Class Diagram*

Class diagram merupakan diagram yang selalu ada di pemodelan system berorientasi objek. Class diagram menunjukkan hubungan antar class dalam system yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai satu tujuan. Kelas pada kelas diagram terdiri dari 3 bagian utama yaitu nama kelas, isi property dari kelas beserta metode yang ada pada kelas tersebut. Kelas juga memiliki jenis-jenis hubungan seperti asosiatif, dependensi, agregasi, komposisi,

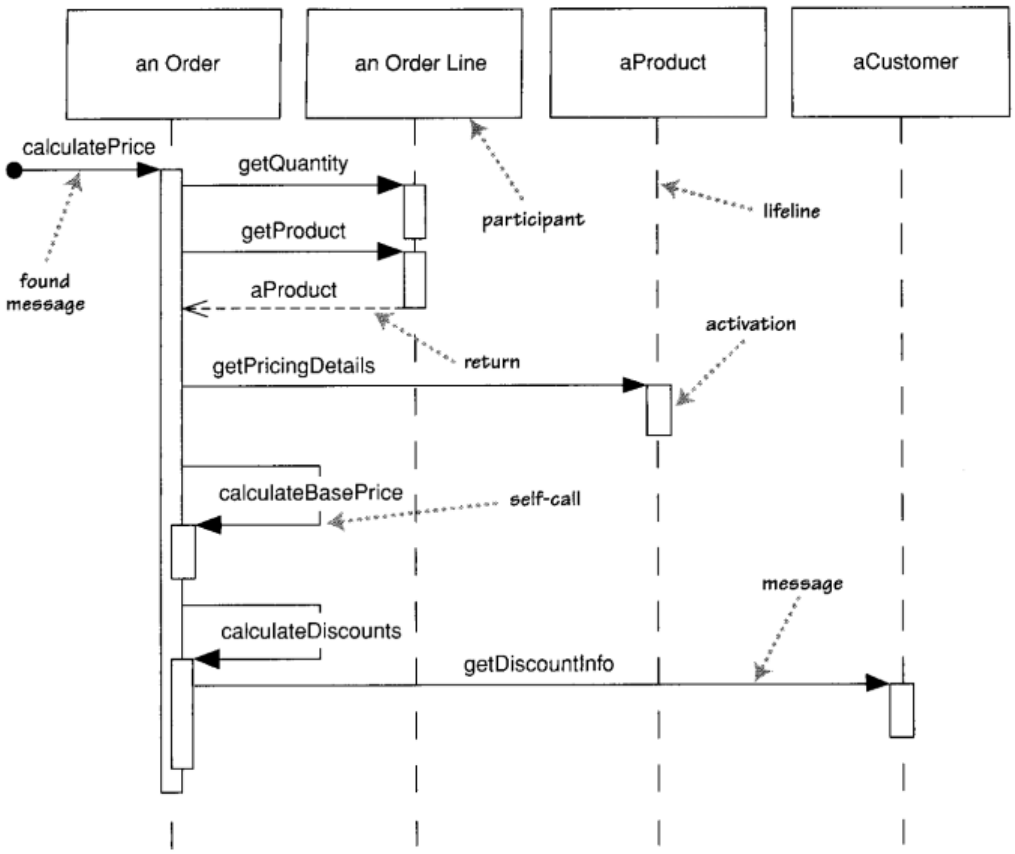
spesifikasi dan generalisasi. Hubungan ini digunakan untuk menggambarkan bagaimana hubungan dan interaksi yang terjadi antar kelas. Masing-masing komponen penyusun kelas memiliki hak akses seperti public, private dan protected [13].



Gambar II.10 Class Diagram [13]

II.2.10.4 *Sequence Diagram*

Sequence Diagram adalah suatu diagram yang digunakan untuk menggambarkan dan menjelaskan secara detil urutan proses yang dilakukan dalam sistem untuk mencapai tujuan dari use case. Adapun urutan proses yang dijelaskan yaitu interaksi yang terjadi antar class, operasi yang terlibat, urutan antar operasi, dan informs yang diperlukan oleh masing-masing operasi. Komponen utamanya adalah objek yang digambarkan dengan kotak segi empat atau bulat, message yang digambarkan dengan garis penuh, dan waktu yang ditunjukkan dengan progress *vertical* [13].



Gambar II.11 Sequence Diagram [13]

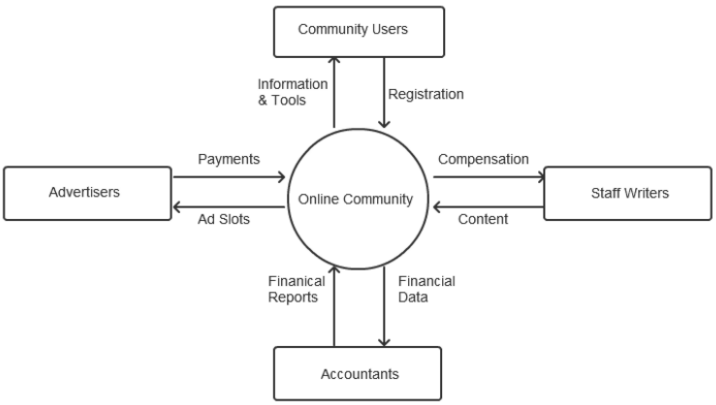
II.2.12 *Structured Analysis and Design*

Structured analysis and design adalah pendekatan sistem analisis dan desain sistem informasi secara hierarki. SADM diproduksi untuk CCTA, kantor pemerintah Inggris yang berkaitan dengan penggunaan teknologi di pemerintahan dari tahun 1980 dan seterusnya [14].

II.2.11.1 *Diagram Konteks*

Diagram Context, adalah data flow diagram tingkat atas (DFD Top Level), yaitu diagram yang paling tidak detail, dari sebuah sistem informasi yang menggambarkan aliran-aliran data ke dalam dan ke luar sistem dan ke dalam dan ke luar entitas-entitas eksternal. (CD menggambarkan sistem dalam satu lingkaran

dan hubungan dengan entitas luar. Lingkaran tersebut menggambarkan keseluruhan proses dalam sistem) [15].



Gambar II.12 Diagram Konteks [16]

II.2.11.2 *Data Flow Diagram*

Data Flow Diagram (DFD) adalah representasi grafik dari sebuah sistem. DFD menggambarkan komponen-komponen sebuah sistem, aliran-aliran data dimana komponen-komponen tersebut, dan asal, tujuan, dan penyimpanan dari data tersebut. DFD terdiri dari empat komponen utama diantaranya adalah:

1) Entitas.
2) Proses.
3) *Data Store.*
4) *Data Flow.*

DFD dapat dikatakan merupakan *breakdown* dari *diagram konteks*. Analisis mengenai sistem yang lebih detail dapat dilakukan dengan melakukan *breakdown* pada DFD level 1, 2 dan seterusnya [15].

II.2.11.3 Spesifikasi Proses

Spesifikasi Proses adalah tabel yang berisi keterangan atau deskripsi dari semua proses yang terdapat pada DFD. Logika proses yang ditulis pada spesifikasi proses harus dituliskan secara jelas baik menggunakan bahasa deskriptif atau *pseudecode* tetapi tidak boleh mengkombinasikan kedua bahasa tersebut [19].

Berikut adalah Tabel II.10 Contoh Spesifikasi Proses:

Tabel II.5 Contoh Spesifikasi Proses

No	Proses	Keterangan
1.	No Proses	Menyatakan no proses
2.	Nama Proses	Menyatakan nama proses
3.	Source (Sumber)	(Menyatakan sumber data input menuju proses)
4.	Input	(menyatakan isi data yang masuk ke proses)
5.	Output	(menyatakan informasi yang keluar dari proses)
6.	Destination	(menatakan tujuan informasi output dari proses)
7.	Logika Proses	(menyatakan algoritma dari proses)

II.2.11.4 Kamus Data

Kamus data adalah tabel yang berisi deskripsi dari data yang mengalir pada DFD [15].

Tabel II.6 Contoh Kamus Data

Nama aliran data	Data Siswa Nama dari data yang digunakan
Where used/How used	Daftar pada poin proses apa data digunakan
Keterangan	Uraian singkat dari data yang digunakan
Struktur Data	Daftar komponen data yang ada pada data yang digunakan
Deskripsi	Jenis data dalam representasi komputer untuk masing-masing data.

II.2.13 *Database*

Database atau basis data adalah kumpulan data yang disimpan secara sistematis di dalam komputer dan dapat diolah atau dimanipulasi menggunakan perangkat lunak (program aplikasi) untuk menghasilkan informasi. Pendefinisian basis data meliputi spesifikasi berupa tipe data, struktur, dan juga batasan-batasan data yang akan disimpan. Basis data merupakan aspek yang sangat penting dalam sistem informasi dimana basis data merupakan gudang penyimpanan data yang akan diolah lebih lanjut. Basis data menjadi penting karena dapat menghidari

duplikasi data, hubungan antar data yang tidak jelas, organisasi data, dan juga update yang rumit.

Proses memasukkan dan mengambil data ke dan dari media penyimpanan data memerlukan perangkat lunak yang disebut dengan sistem manajemen basis data (database management system | DBMS). DBMS merupakan sistem perangkat lunak yang memungkinkan user untuk memelihara, mengontrol, dan mengakses data secara praktis dan efisien. Dengan kata lain semua akses ke basis data akan ditangani oleh DBMS. Ada beberapa fungsi yang harus ditangani DBMS yaitu mengolah pendefinisian data, dapat menangani permintaan pemakai untuk mengakses data, memeriksa sekuriti dan integriti data yang didefinisikan oleh DBA (Database Administrator), menangani kegagalan dalam pengaksesan data yang disebabkan oleh kerusakan sistem maupun disk, dan menangani unjuk kerja semua fungsi secara efisien.

Tujuan utama dari DBMS adalah untuk memberikan tinjauan abstrak data kepada user (pengguna). Jadi sistem menyembunyikan informasi tentang bagaimana data disimpan, dipelihara, dan tetap dapat diambil (akses) secara efisien. Pertimbangan efisien disini adalah bagaimana merancang struktur data yang kompleks tetapi masih tetap bisa digunakan oleh pengguna awam tanpa mengetahui kompleksitas strukturnya. Database dibagi menjadi dua jenis yaitu [17]:

1. Basis Data Flat File

Basis data flat-file ideal untuk data berukuran kecil dan dapat dirubah dengan mudah. Pada dasarnya, mereka tersusun dari sekumpulan string dalam satu atau lebih file yang dapat diurai untuk mendapatkan informasi yang disimpan. Basis data flat-file baik digunakan untuk menyimpan daftar atau data yang sederhana dan dalam jumlah kecil. Basis data flat-file akan menjadi sangat rumit apabila digunakan untuk menyimpan data dengan struktur kompleks walaupun dimungkinkan pula untuk menyimpan data semacam itu. Salah satu masalah menggunakan basis data jenis ini adalah rentan pada korupsi data karena tidak adanya penguncian yang melekat ketika data digunakan atau dimodifikasi.

2. Basis Data Relasional

Basis data ini mempunyai struktur yang lebih logis terkait cara penyimpanan. Kata "relasional" berasal dari kenyataan bahwa tabel-tabel yang berada di basis data dapat dihubungkan satu dengan lainnya. Basis data relasional menggunakan sekumpulan tabel dua dimensi yang masing-masing tabel tersusun atas baris (tupel) dan kolom (atribut). Untuk membuat hubungan antara dua atau lebih tabel, digunakan key (atribut kunci) yaitu **primary key** di salah satu tabel dan **foreign key** di tabel yang lain. Saat ini, basis data relasional menjadi pilihan karena keunggulannya. Beberapa kelemahan yang mungkin dirasakan untuk basis data jenis ini adalah implementasi yang lebih sulit untuk data dalam jumlah besar dengan tingkat kompleksitasnya yang tinggi dan proses pencarian informasi yang lebih lambat karena perlu menghubungkan tabel-tabel terlebih dahulu apabila datanya tersebar di beberapa tabel.

II.2.14 *MySQL*

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL (bahasa Inggris: *database management system*) atau DBMS yang *multithread*, *multi-user*, dengan sekitar 6 juta instalasi di seluruh dunia. *MySQL* AB membuat *MySQL* tersedia sebagai perangkat lunak gratis di bawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus dimana penggunaannya tidak cocok dengan penggunaan GPL [18].

Tidak seperti PHP atau Apache yang merupakan software yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh penulisnya masing-masing, *MySQL* dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia yaitu *MySQL* AB. *MySQL* AB memegang penuh hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan *MySQL* AB adalah: David Axmark, Allan Larsson, dan Michael "Monty" Widenius. Adapun fitur-fitur *MySQL* antara lain [12]:

1. *Relational Database System*. Seperti halnya *software database* lain yang ada di pasaran, MySQL termasuk RDBMS.
2. Arsitektur *Client-Server*. MySQL memiliki arsitektur *client-server* dimana *server database* MySQL terinstal di *server*. *Client* MySQL dapat berada di

komputer yang sama dengan *server*, dan dapat juga di komputer lain yang berkomunikasi dengan *server* melalui jaringan bahkan *internet*.

3. Mengenal perintah SQL standar. SQL (*Structured Query Language*) merupakan suatu bahasa standar yang berlaku di hampir semua software *database*. MySQL mendukung SQL versi SQL:2003.
4. Mendukung Sub Select. Mulai versi 4.1 MySQL telah mendukung select dalam select (*sub select*).
5. Mendukung *Views*. MySQL mendukung *views* sejak versi 5.0
6. Mendukung *Stored Prosedured* (SP). MySQL mendukung SP sejak versi 5.0.
7. Mendukung *Triggers*. MySQL mendukung *trigger* pada versi 5.0 namun masih terbatas. Pengembang MySQL berjanji akan meningkatkan kemampuan trigger pada versi 5.1.
8. Mendukung *replication*.
9. Mendukung transaksi.
10. Mendukung *foreign key.*

Perintah-perintah yang ada di SQL antara lain:

1. Menambah data (*Insert*)

   Perintah SQL yang digunakan untuk menambah data ke *database* adalah perintah *INSERT*. Perintah ini mempunyai bentuk umum sebagai berikut:

   INSERT [INTO] nama_tabel [(daftar *field*/kolom)] VALUES (daftar_data).
2. Mengakses data (*Select*)

   Perintah SQL yang digunakan untuk memilih data adalah perintah *SELECT*. Bentuk umum perintah ini adalah sebagai berikut:

   SELECT daftar_select *FROM* daftar_tabel

   [WHERE kondisi_pencarian]

   [GROUP BY daftar_group_by]

   [HAVING kondisi_pencarian]

   [ORDER BY daftar_order[ASC │DESC]]
3. Mengedit/Mengubah data (*Update*)

   Perintah SQL yang digunakan untuk mengubah data adalah perintah *UPDATE SET*. Perintah ini mempunyai bentuk umum sebagai berikut:

UPDATE nama_table SET field1=databaru1
[data2=databaru2]
[WHERE kondisi_update].

4. Menghapus data (*Delete*)
   Perintah SQL yang digunakan untuk menghapus data adalah perintah *DELETE FROM*. Perintah ini mempunyai bentuk umum sebagai berikut:
   DELETE FROM tabel_hapus WHERE kondisi_hapus.

II.2.15 Internet

Internet (*interconnection networking*) adalah jaringan komunikasi global yang terbuka dan menghubungkan jutaan bahkan milyaran jaringan komputer dengan berbagai tipe dan jenis, dengan menggunakan tipe komunikasi seperti telepon, satelit dan lain sebagainya. Awalnya internet merupakan jaringan komputer yang dibentuk oleh Departemen Pertahanan Amerika Serikat pada tahun 1969 melalui sebuah proyek yang disebut dengan ARPANET. Misi awal dari proyek ini awalnya hanya untuk keperluan militer saja, tetapi lambat laun terus berkembang dan bisa dinikmati oleh semua kalangan. Terciptanya internet telah membawa perubahan yang sangat berarti dalam berbagai aspek kehidupan manusia. Selain itu, internet juga telah melahirkan dunia baru yang memiliki pola, corak sekaligus karakteristik yang berbeda dengan dunia nyata [19].

Internet yang sering disebut dengan jagad raya informasi menyajikan sekian banyak fasilitas yang dapat dinikmati oleh pengguna internet. Berikut ini beberapa fasilitas yang dapat dinikmati oleh pengguna internet yaitu www (*world wide web*), email, *mailing list,* BBS, *chatting, newsgroup,* FTP dan *gopher* [24].

II.2.16 Bootstrap

Bootstrap merupakan sebuah *framework* CSS dari Twitter, yang menyediakan kumpulan komponen-komponen antarmuka dasar pada web yang telah dirancang sedemikian rupa untuk digunakan bersama-sama. Selain komponen antarmuka, Bootstrap juga menyediakan sarana untuk membangun *layout* halaman dengan mudah dan rapi, serta modifikasi pada tampilan dasar HTML untuk

membuat seluruh halaman web yang dikembangkan senada dengan komponen-komponen lainnya.

Lisensi yang digunakan oleh Bootstrap yaitu lisensi Apache 2.0, sebuah lisensi yang sangat terbuka sehingga kita dapat dengan bebas dan mudah menggunakan Bootstrap, tanpa perlu khawatir akan ancaman legal dari Twitter atau pihak lainnya.



Gambar II.13 Bootstrap [20]