BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

Landasan teori adalah sekumpulan definisi atau pengertian dan gambaran dari konsep yang telah disusun secara sistematis dari penelitian yang sedang diteliti dan menjadikannya sebagai landasan yang kuat atas penelitian yang sedang berjalan. Landasan teori yang berkaitan dengan materi atau teori yang digunakan sebagai acuan melakukan penelitian. Landasan teori yang diuraikan merupakan hasil studi literatur, buku-buku, maupun situs internet.

2.1.1 Jasa

Jasa memiliki arti pemberian suatu kinerja atau tindakan tak kasat mata dari satu pihak kepada pihak lain dalam hal penelitian ini dari pihak penyedia jasa tukang kepada pihak pengguna jasa tukang. Pada umumnya jasa diproduksi dan dikonsumsi secara bersamaan, di mana interaksi antara pemberi jasa dan penerima jasa mempengaruhi kerja tersebut [11].

Jasa juga merupakan suatu kegiatan yang dapat diidentifikasikan, yang bersifat tak teraba, yang direncanakan untuk pemenuhan kepuasan konsumen. Untuk menghasilkan jasa mungkin perlu atau mungkin juga tidak perlu penggunaan barang yang berwujud, akan tetapi tidak terdapat pemindahan hak milik atas benda tersebut [12].

Majunya kehidupan di segala bidang mempengaruhi jasa yang dibutuhkan manusia. Secara garis besar macam jasa yang dibutuhkan manusia bisa diklasifikasikan atas beberapa macam, yakni:

1. Perumahan (termasuk sewa kamar hotel, motel, apartemen/rumah flat, usaha tani, dan lain-lain).
2. Usaha rumah tangga (termasuk air minum, perbaikan rumah, reparasi alat rumah tangga, perawatan kebun, pembersihan, pertukangan, dan lain-lain).

3. Rekreasi dan kesukaan (penyewaan dan reparasi peralatan untuk ikut serta dalam kegiatan rekreasi dan hiburan, juga izin memasuki gelanggang hiburan, rekreasi dan kesenangan lainnya).
4. Perawatan pribadi (binatu pakaian, dan perawatan kecantikan).
5. Perawatan medis dan kesehatan (perawatan gigi, perawatan sakit, opname di rumah sakit, dan periksa dokter).
6. Pendidikan privat dan kursus-kursus.
7. Jasa bisnis dan profesi lainnya (jasa hukum, akuntan, konsultasi manajemen, dan jasa komputer).
8. Asuransi, bank, dan jasa finansial lainnya (asuransi pribadi dan bisnis, jasa kredit dan pinjaman, konsultasi investasi, dan pajak).
9. Transportasi (jasa angkutan barang dan penumpang, reparasi, dan penyewaan mobil).
10. Komunikasi (telepon, telegram, dan komputer).

2.1.2 Tukang

Tukang adalah sebuah homonim sebab arti-artinya memiliki ejaan dan pelafalan yang sama tetapi maknanya berbeda. Tukang memiliki arti dalam kelas nomina atau kata benda sehingga tukang dapat menyatakan nama dari seseorang, tempat, atau semua benda dan segala yang dibendakan. Tukang termasuk dalam ragam bahasa cakapan. Tukang memiliki 5 arti [13].

1. Tukang berarti orang yang mempunyai kepandaian dalam suatu pekerjaan tangan (dengan alat atau bahan yang tertentu)
   Contoh : Tukang batu, tukang besi, tukang kayu.
2. Tukang berarti orang yang pekerjaannya membuat (menjual, memperbaiki, dan sebagainya) sesuatu yang tentu.
3. Tukang berarti orang yang pekerjaannya melakukan sesuatu secara tetap.
   Contoh: Tukang pangkas (cukur), tukang las, tukang jahit, tukang masak, tukang cetak.
4. Tukang berarti orang yang biasa suka melakukan sesuatu (yang kurang baik).

Contoh: Tukang mabuk, tukang serobot, tukang copet, tukang tadah, tukang catut.

5. Tukang berarti ahli (untuk mencemoohkan).

   Contoh: Tukang menciptakan sajak, tukang pidato

Dalam penelitian ini yang lebih spesifik tukang berarti orang yang mempunyai kepandaian dalam suatu pekerjaan tangan seperti perbaikan elektronik, kelistrikan, dan perbaikan peralatan rumah.

2.1.3 Jasa Tukang

Jasa tukang termasuk Usaha Mikro, Kecil dan Menengah pada bidang jasa. Jasa tukang merupakan sebuah layanan publik yang merupakan orang yang mempunyai kepandaian dalam suatu pekerjaan tangan seperti perbaikan elektronik, kelistrikan, dan perbaikan peralatan rumah dan merupakan sebuah kegiatan atau manfaat yang di tawarkan oleh suatu pihak penyedia jasa tukang ke pihak pengguna jasa tukang [1], kegiatan jasa tukang tersebut bisa berarti memperbaiki atau mereparasi suatu objek atau barang-barang yang butuh untuk diperbaiki. Jasa tukang yang dimaksud dalam penelitian ini adalah jasa tukang perbaikan elektronik, kelistrikan, dan perbaikan peralatan rumah.

2.1.4 Pemesanan

Pemesanan adalah suatu aktivitas yang dilakukan oleh konsumen sebelum melakukan pemesanan jasa, tempat ataupun suatu barang. Untuk mewujudkan kepuasan konsumen maka diperlukan adanya sebuah sistem pemesanan yang baik. Menurut kamus besar bahasa Indonesia yang dimaksud pemesanan adalah proses, perbuatan, cara memesan (tempat, jasa, barang, dsb.) kepada orang lain [13].

2.1.5 *On Demand*

On demand merupakan bahasa Inggris yang apabila diartikan ke dalam bahasa Indonesia berarti “sesuai permintaan”, dalam penelitian ini dimuat kata *on demand* berarti agar dalam proses pemesanan dapat sesuai permintaan, permintaan itu mencakup permintaan keahlian, kebutuhan, dan permintaan jadwal yang sesuai.

2.1.6 Aplikasi

Aplikasi merupakan perangkat lunak yang dijalankan oleh para pengguna atau biasa disebut dengan *user* untuk mendapat suatu tujuan tertentu. Aplikasi perangkat lunak adalah suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung yang bertujuan untuk melakukan suatu tugas yang diinginkan pengguna. Aplikasi perangkat lunak adalah program yang membuat komputer dapat digunakan untuk pekerjaan sehari-hari agar lebih efektif dan efisien

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 dirasakan kegunaannya 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 aplikasi kalkulator yang sederhana atau serumit aplikasi pengolah kata.

2.1.6.1 Aplikasi Mobile

Aplikasi Mobile adalah perangkat lunak yang berjalan di perangkat *mobile* seperti *smartphone* atau tablet PC. Aplikasi Mobile juga dikenal sebagai aplikasi yang dapat diunduh dan memiliki fungsi tertentu sehingga menambah fungsionalitas dari perangkat *mobile* itu sendiri. Untuk mendapatkan *mobile application* yang diinginkan, *user* dapat mengunduhnya melalui situs tertentu sesuai dengan sistem operasi yang dimiliki. Google Play dan iTunes merupakan beberapa contoh dari situs yang menyediakan beragam aplikasi bagi pengguna Android dan iOS untuk mengunduh aplikasi yang diinginkan [15].

2.1.7 Android

Android merupakan sebuah sistem operasi untuk perangkat *mobile* berbasis linux yang mencakup sistem operasi, *middleware* dan aplikasi. Android menyediakan platform terbuka bagi para pengembang untuk menciptakan aplikasi mereka [16]. Awalnya, Google Inc. membeli Android Inc. yang merupakan pendatang baru yang membuat peranti lunak untuk ponsel/*smartphone*. Kemudian untuk mengembangkan Android, dibentuklah Open Handset Alliance, konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcomm, T-Mobile, dan Nvidia.

Pada saat perilisan perdana Android, 5 November 2007, Android bersama Open Handset Alliance menyatakan mendukung pengembangan open *source* pada perangkat *mobile*. Di lain pihak, Google merilis kode-kode Android di bawah lisensi Apache, sebuah lisensi perangkat lunak dan open platform perangkat seluler.

Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari Google atau *Google Mail Service* (GMS) dan kedua adalah yang benar-benar bebas distribusinya tanpa dukungan langsung Google atau dikenal sebagai *Open Handset Distribution* (OHD).

Sekitar September 2007 Google mengenalkan Nexus One, salah satu jenis *smartphone* yang menggunakan Android sebagai sistem operasinya. Telepon seluler ini diproduksi oleh HTC Corporation dan tersedia di pasaran 5 Januari 2010. Pada 9 Desember 2008, diumumkan anggota baru yang bergabung dalam program kerja Android ARM Holdings, Atheros, Communications, diproduksi oleh Asustek Computer Inc, Garmin Ltd, Softbank, Sony Ericsson, Toshiba Corp, dan Vodafone Gropu Plc. Seiring pembentukan Open handset Alliance, OHA mengumumkan produk perdana mereka, Android, perangkat mobile yang merupakan modifikasi kernel Linux 2.6. Sejak Android dirilis telah dilakukan berbagai pembaruan berupa perbaikan *bug* dan penambahan fitur baru.

Pada masa saat ini kebanyakan vendor-vendor *smartphone* sudah memproduksi *smartphone* berbasis Android, vendor-vendor itu antara lain HTC, Motorola, Samsung, LG, HKC, Huawei, Archos, Webstation Camangi, Dell,

Nexus, SciPhone, WayteQ, Sony Ericcson, LG, Acer, Philips, T-Mobile, Nexian, IMO, Asus dan masih banyak lagi vendor *smartphone* didunia yang memproduksi Android. Hal ini karena Android itu adalah sistem operasi yang open *source* sehingga bebas didistribusikan dan dipakai oleh vendor mana pun.

Tidak hanya menjadi sistem operasi di *smartphone*, saat ini Android menjadi pesaing utama dari Apple pada sistem operasi *Table* PC, pesatnya pertumbuhan Android selain faktor yang disebutkan di atas adalah karena Android itu sendiri adalah platform yang sangat lengkap baik itu sistem operasinya, Aplikasi dan *Tool* Pengembangan, *Market* aplikasi Android serta dukungan yang sangat tinggi dari komunitas *Open Source* di dunia, sehingga Android terus berkembang sangat pesat baik dari segi teknologi maupun dari segi jumlah *device* yang ada di dunia [16].

2.1.7.1 Arsitektur Android

Secara garis besar arsitektur android dapat dijelaskan dan digambarkan sebagai berikut [16].

1. *Applications* dan *Widgets*

   Applications dan *Widgets* ini merupakan layer di mana kita hanya berhubungan dengan aplikasi saja, di mana biasanya *download* aplikasi kemudian dilakukan instalasi dan jalankan aplikasi tersebut. Di layer terdapat aplikasi inti termasuk klien email, program SMS, kalender, peta, browser, kontak, dan lain-lain. Semua aplikasi ditulis menggunakan bahasa pemrograman Java.

2. *Applications Frameworks*

   Android merupakan *Open Development Platform* yaitu Android menawarkan kepada pengembang atau memberi kemampuan kepada pengembang untuk membangun aplikasi yang baik dan inovatif. Pengembang bebas untuk mengakses perangkat keras, akses informasi *resources*, menjalankan *service background*, mengatur alarm, dan menambahkan status *notifications*, dan sebagainya. Pengembang mempunyai akses ke API *framework* seperti yang dilakukan oleh aplikasi

yang memiliki kategori inti. Arsitektur aplikasi dirancang supaya kita dengan mudah dapat menggunakan kembali komponen yang sudah digunakan (*reuse*).

Oleh karena itu bisa disimpulkan *Applications Frameworks* ini adalah layer di mana para *programmer* melakukan pengembangan ataupun pembuatan aplikasi yang akan dijalankan di sistem operasi Android, karena pada layer inilah aplikasi bisa dirancang dan dibuat, seperti *content providers* yang berupa sms dan panggilan telepon.

Komponen-komponen yang termasuk di dalam *Applications Frameworks* adalah sebagai berikut:

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

3. *Libraries*

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

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

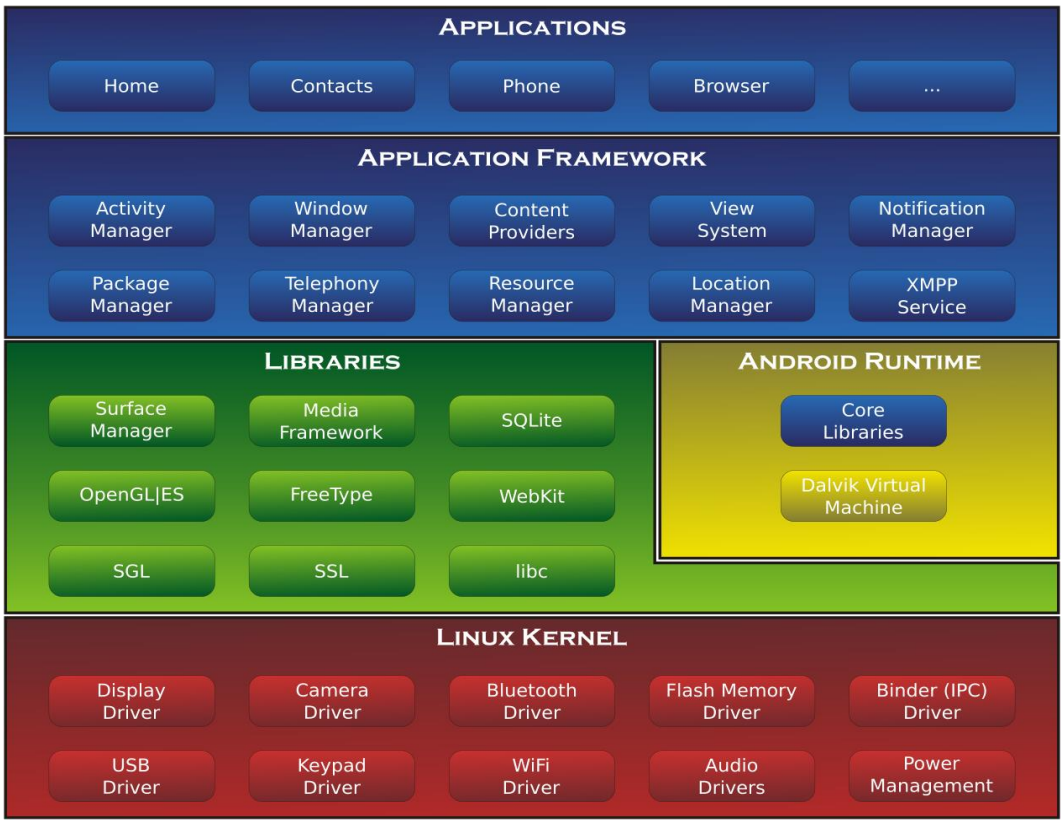
4. Android *Run Time*

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

   a. Core Libraries: Aplikasi Android dibangun dalam bahasa Java, sementara Dalvik sebagai virtual mesinnya bukan Virtual Machine Java, sehingga diperlukan sebuah *libraries* yang berfungsi untuk menerjemahkan bahasa Java/C ditangani oleh Core Libraries.
   b. Dalvik Virtual Machine: Virtual mesin berbasis register yang dioptimalkan untuk menjalankan fungsi-fungsi secara efisien, di mana merupakan pengembangan yang mampu membuat linux kernel untuk melakukan *threading* dan manajemen tingkat rendah.

5. Linux Kernel

   Adalah layer di mana inti dari sistem operasi dari Android itu berada. Berisi *file-file* sistem yang mengatur sistem *processing, memory, resource, drivers,* dan sistem-sistem operasi Android lainnya. Linux kernel yang digunakan Android adalah linux kernel release 2.6

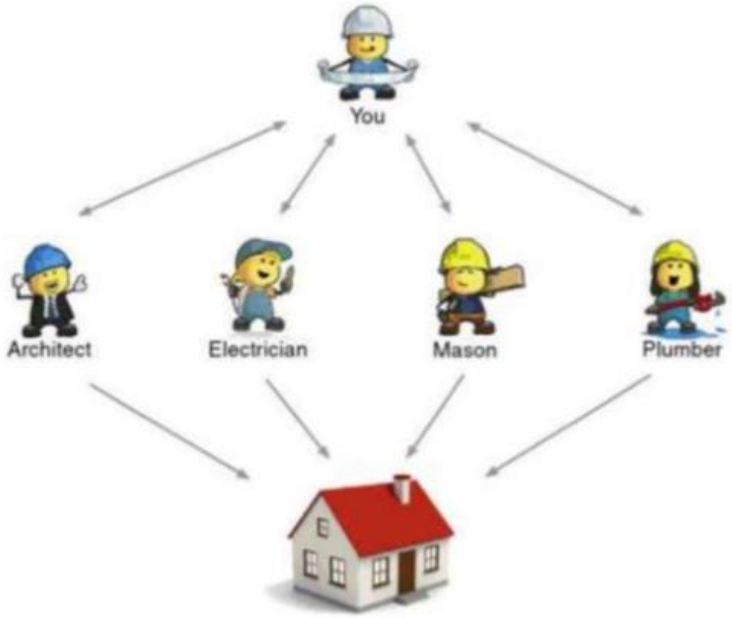


Sumber Gambar : https://id.wikipedia.org/wiki/Berkas:Android-System-Architecture.svg

Gambar 0.1 Arsitektur Android

2.1.8 API (*Application Programming Interface*)

Application Programming Interface (API) merupakan *software interface* yang terdiri dari kumpulan instruksi yang disimpan dalam bentuk *library* dan menjelaskan bagaimana agar suatu *software* dapat berinteraksi dengan *software* lain. Penjelasan ini dapat dicontohkan dengan analogi apabila akan dibangun suatu bangunan atau rumah. Dengan menyewa kontraktor yang dapat menangani bagian yang berbeda, pemilik rumah dapat memberikan tugas yang perlu dilakukan oleh kontraktor tanpa harus mengetahui bagaimana cara kontraktor menyelesaikan pekerjaan tersebut. Dari analogi tersebut, rumah merupakan *software* yang akan dibuat, dan kontraktor merupakan API yang mengerjakan bagian tertentu dari *software* tersebut tanpa harus diketahui bagaimana prosedur dalam melakukan pekerjaan tersebut [17]. Analogi API dapat dilihat pada gambar 2.1 berikut:



Sumber Gambar : Jurnal Teknologi Terpadu [17]**.**

Gambar 0.2 Analogi API

2.1.9 Bahasa Pemrograman

Bahasa pemrograman merupakan kumpulan aturan yang disusun sedemikian rupa sehingga memungkinkan pengguna komputer membuat program yang dapat dijalankan dengan aturan tersebut. Bahasa pemrograman dapat dikelompokkan dalam berbagai macam sudut pandang [18]. Salah satu pengelompokan bahasa pemrograman adalah pendekatan dari notasi bahasa pemrograman tersebut, apakah lebih dekat ke bahasa mesin atau bahasa manusia. Dengan cara ini, bahasa pemrograman dapat dikelompokkan menjadi tiga yakni:

1. *Low Level Language*

   Yaitu bahasa pemrograman tingkat rendah, merupakan bahasa pemrograman generasi pertama. Bahasa pemrograman generasi pertama ini merupakan bahasa pemrograman yang sangat sulit dimengerti karena instruksinya menggunakan bahasa mesin, dan hanya dimengerti oleh pembuatnya saja karena programnya berupa kode-kode mesin. Bahasa pemrograman level rendah ini pertama kali muncul atau digunakan mulai sekitar tahun 1945. Ketika itu untuk membuat dan menjalankan suatu program dibutuhkan waktu yang lama dan itu pun sering dijumpai

kesalahan-kesalahan. Program sangat sulit dibaca dan dibaca, sehingga pada saat itu sangat sedikit orang yang tertarik untuk menjadi *programmer*.

2. *Middle Level Language*

   Yaitu bahasa pemrograman tingkat menengah, yang merupakan bahasa pemrograman generasi ke dua. Dalam bahasa ini seorang *programmer* sudah mulai bisa menggunakan bahasa sehari-hari, walaupun masih banyak susah dimengerti juga. Banyak perintah-perintah yang menggunakan inisial atau singkatan-singkatan seperti "MOV" yang berarti "MOVE" (pindah), "STO" yang berarti (STORE) dan lain-lain. Bahasa pemrograman yang tergolong dalam *middle level* ini adalah *assembler*.

3. *High Level Language*

   Yaitu bahasa pemrograman tingkat tinggi, yang merupakan bahasa pemrograman generasi ke tiga dan selanjutnya. Ciri-cirinya yaitu bahasa pemrograman ini sudah terstruktur dengan baik, mudah dimengerti karena sudah menggunakan bahasa sehari-hari(–bahasa inggris tapi ya–), Bahasa pemrograman inilah bahasa pemrograman yang sekarang ini kita kenal, seperti C, C++, JAVA, PHP, Visual Basic, Pascal, ORACLE, MS-SQL, Phyton, XML dan lain-lain sebagainya. Sudah begitu beragam dan bermacam-macam jenis sesuai karakter struktur dan kegunaannya.

2.1.10 *Hybrid*

2.1.10.1 *Hybrid Apps*

Hybrid apps merupakan sebutan untuk aplikasi *smartphone* yang dibuat dengan teknologi web. Dikatakan *hybrid apps* karena aplikasi tersebut merupakan kombinasi antara aplikasi web dan aplikasi *native*. Aplikasi web merupakan aplikasi yang berada di server web dan diakses menggunakan web browser, sedangkan aplikasi *native* merupakan aplikasi yang dibuat dengan bahasa asli sistem operasi. *Hybrid apps* juga menggabungkan keuntungan yang ada pada

aplikasi web yang *multi platform* dan keuntungan *native apps* yang mampu mengakses fitur-fitur *native* pada *smartphone* [19].

Dibanding dengan aplikasi *native*, aplikasi *hybrid* memiliki banyak keunggulan. Keunggulan aplikasi *hybrid* dibandingkan dengan aplikasi *native* yaitu:

1. Cukup membuat aplikasi satu kali untuk digunakan oleh berbagai sistem operasi *smartphone*.
2. Menggunakan bahasa yang mudah dipahami yaitu CSS3, HTML5 dan javascript yang sudah sangat dikenal bagi web *programmer*.
3. Biaya pengembangan lebih rendah.
4. Waktu yang diperlakukan untuk pengembangan aplikasi lebih sedikit.

Adapun bila dibandingkan dengan aplikasi web, aplikasi *hybrid* memiliki banyak keunggulan sebagai berikut:

1. Proses *loading* dan kinerja lebih cepat.
2. Dapat dijalankan secara *online* maupun *offline*.
3. Dapat mengakses fitur-fitur perangkat keras *smartphone*, seperti kamera, galeri, kontak, dan sebagainya.
4. Dapat di *upload* ke *appstore*.

2.1.10.2 Node.js

Node.js merupakan perangkat lunak yang didesain untuk mengembangkan aplikasi berbasis web dan ditulis dalam sintak bahasa pemrograman JavaScript. Bila selama ini kita mengenal JavaScript sebagai bahasa pemrograman yang berjalan di sisi *client* / browser saja, maka Node.js ada untuk melengkapi peran JavaScript sehingga bisa juga berlaku sebagai bahasa pemrograman yang berjalan di sisi server, seperti halnya PHP, Ruby, Perl, dan sebagainya. Node.js dapat berjalan di sistem operasi Windows, Mac OS X dan Linux tanpa perlu ada perubahan kode program. Node.js memiliki pustaka server HTTP sendiri sehingga memungkinkan untuk menjalankan server web tanpa menggunakan program server web seperti Apache atau Nginx [19].

Kelebihan Memakai Node.js:

1. Node.js menggunakan bahasa pemrograman JavaScript yang merupakan bahasa pemrograman yang paling populer dan banyak dikenal oleh masyarakat luas
2. Node.js mampu menangani ribuan koneksi bersamaan dengan penggunaan *resource* minimum untuk setiap prosesnya
3. Node.js sangat diandalkan terutama untuk membuat aplikasi real-*time*
4. Node.js adalah *project open source*, sehingga siapa pun dapat melihat struktur kode dan juga dapat berkontribusi untuk pengembangannya
5. Penggunaan JavaScript di sisi server dan juga *client* meminimalisir ketidakcocokan antar dua sisi lingkungan pemrograman, seperti terkait komunikasi data yang mana menggunakan struktur JSON yang sama di kedua sisi, validasi *form* yang sama yang dapat dijalankan di sisi server dan *client*, dan sebagainya.
6. Database NoSQL seperti MongoDB dan CouchDB mendukung langsung Javascript sehingga *interfacing* dengan *database* ini akan jauh lebih mudah.
7. Node.js memakai V8 yang selalu mengikuti perkembangan standar ECMAScript (nama standar resmi dari JavaScript, Namun JavaScript yang lebih dikenal dalam implementasinya), sehingga tidak perlu ada kekhawatiran bahwa browser tidak mendukung fitur-fitur di Node.js.

2.1.10.2.1 NPM (*Node Package Manager*)

Salah satu *tool* yang akan sering digunakan dalam Nodejs adalah NPM (*Node Package Manager*) [19]. NPM sudah otomatis terinstal saat menginstal Nodejs. NPM berfungsi untuk:

1. Membuat Project Baru;
2. Menginstal modul atau *library*;
3. Menjalankan skrip *command line.*

2.1.10.3 *Ionic Framework dan Apache Cordova*

Framerwork merupakan kumpulan kode program yang siap pakai sehingga ketika membuat aplikasi tidak perlu dimulai dari nol. Jika dalam aplikasi web dikenal dengan Bootstrap sebagai *framework* populer yang menyediakan berbagai desain siap pakai dan terintegrasi dengan jQuery. Namun tidak cocok ketika digunakan untuk aplikasi *hybrid* [19].

Ionic merupakan kerangkan ponsel HTML5 dengan fokus pada kinerja yang memanfaatkan akselerasi *hardware* dan tidak memerlukan pihak ketiga seperti JS *library*. Ionic bekerja berbarengan dengan Angular.js untuk membangun sebuah aplikasi interaktif. Aplikasi *hybrid* pada dasarnya ialah *website* kecil yang berjalan di *shell* browser, sebuah aplikasi yang mempunyai akses ke lapisan platform asli dari sebuah *device*. Aplikasi *hybrid* mempunyai sangat banyak manfaat jika dibandingkan dengan aplikasi *native*, khususnya dalam hal mendukung platform dan kecepatan pengembangan, Ionic dilengkapi dengan elemen UI *mobile* dan *layout* yang mirip dengan SDK asli pada ios atau android. Ionic memerlukan Apache Cordova untuk menjalankan aplikasi [20].

Dalam pengembangan aplikasi *hybrid*, juga banyak pihak yang menawarkan *framework* dengan segala kemampuannya yang akan sangat membantu pembuatan aplikasi menjadi lebih mudah. Salah satu yang paling populer dan banyak digunakan oleh para *programmer* di dunia yaitu *framework* Ionic. Ionic terintegrasi dengan Angular sehingga sangat cocok digunakan untuk membuat aplikasi yang berbasis Angular.

Sejak versi 2, Angular sudah banyak berbeda dibanding dengan versi pertamanya. Untungnya, Ionic merupakan *framework* yang selalu *update* dan mengikuti perkembangan Angular.

Beberapa keunggulan yang dimiliki Ionic dibanding dengan para pesaingnya di antaranya sebagai berikut:

1. Ionic berbasis open *source* sehingga bebas mengembangkan aplikasi baik untuk keperluan sendiri maupun aplikasi komersial menggunakan Ionic.
2. Ionic terintegrasi dengan Angular, bahkan gaya pengkodeannya juga persis mengikuti gaya pengkodean Angular.

3. Ionic memilik UI *default* yang sangat cantik dan mudah untuk di *customize*.
4. Ionic dapat di instal melalui GUI maupun CLI, serta menyediakan *Tool* dan Service yang memudahkan pengguna Ionic bagi *programmer*.
5. Ionic dapat diintegrasikan dengan mudah dengan fitur-fitur *smartphone* seperti kamera, galeri, kontak, *geolocation*, dan sebagainya.
6. Tim pengembang Ionic sangat aktif di media sosial.
7. Ionic juga memiliki komunitas yang sangat besar aktif di media sosial.

Apache Cordova adalah *framework* untuk membangun sebuah aplikasi pada berbagai macam platform seperti Android, Blackberry, Iphone ataupun Windows Phone menggunakan HTML5, JQuery, JQuery Mobile dan CSS3. Membangun aplikasi untuk *device* yang berbeda seperti Android, iPhone, Windows Mobile dan lainnya dibutuhkan *framework* dan bahasa pemrograman yang berbeda, seperti pada Android menggunakan bahasa pemrograman Jawa, blackberry dengan bahasa pemrograman Jawa, iPhone dengan Basic C dan Windows Phone dengan C# [20].

2.1.11 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 [21]. 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 pemahaman, 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 sistem.

Berikut adalah contoh sintak bahasa pemrograman PHP:

```
<!DOCTYPE html>
<html>
    <head>
        <title><?php echo "Belajar PHP" ?></title>
    </head>
    <body>
        <?php
            echo "saya sedang belajar PHP<br>";
            echo "<p>Belajar PHP hingga jadi master</p>";
        ?>
    </body>
</html>
```

Gambar 0.3 Contoh Sintak PHP

2.1.12 *Database*

Database adalah sekumpulan tabel-tabel yang berisi data dan merupakan kumpulan dari *field* atau kolom. Struktur *file* yang menyusun sebuah *database* adalah data *Record* dan *Field* [22].

1. Data adalah satu kesatuan informasi yang akan diolah. Sebelum diolah, data dikumpulkan di dalam suatu *file database*.

2. *Record* adalah data yang isinya merupakan satu kesatuan seperti NamaUser dan *Password* dinamakan satu *record*. Setiap *record* diberi nomor urut yang disebut nomor *record* (*Record Number*).
3. *Field* adalah sub bagian dari *Record*. Dari contoh isi *record* di atas, maka terdiri dari 2 *field*, yaitu: *field* NamaUser dan *Password*.

2.1.12.1 MySQL

MySQL adalah sebuah perangkat lunak sistem manajemen basis data SQL atau yang dikenal dengan DBMS (*Database Management System*), *database* ini *multithread, multi-user*. MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis di bawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual di bawah lisensi komersial untuk kasus-kasus yang bersifat khusus.

Kekuatan MySQL tidak ditopang oleh sebuah komunitas, seperti Apache, yang dikembangkan oleh komunitas umum, dan hak cipta untuk kode sumber dimiliki oleh pemilik masing-masing, tetapi MySQL didukung penuh oleh sebuah perusahaan profesional dan komersial, yakni MySQL AB dari Swedia.

MySQL adalah *Relational Database Management System* (RDBMS) yang didistribusikan secara gratis di bawah lisensi GPL (*General Publi License*). Di mana setiap orang bebas untuk menggunakan MySQL, namun tidak boleh dijadikan produk turunan yang bersifat *closed source* atau komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian *database*, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan mudah secara otomatis [23].

Berikut adalah contoh sintak sederhana pada MySQL:

1. Membuat *database*

```
CREATE DATABASE rental;
```

2. Melihat *database*

```
SHOW DATABASES;
```

3. Membuat tabel

```
CREATE TABLE customer (
        id_customer VARCHAR (8) NOT NULL,
        nama_customer VARCHAR (20) NOT NULL,
        tgl_rental DATETIME NOT NULL,
        tgl_kembali DATETIME NOT NULL,
        PRIMARY KEY (id)
);
```

4. Melihat tabel

```
SHOW TABLES;
```

5. Menghapus tabel

```
DROP TABLE customer;
```

6. Menghapus database

```
DROP DATABASE rental;
```

Dalam menggunakan *database* MySQL, setiap perintah yang diketikkan disebut *query*. Perintah MySQL dapat dikategorikan menjadi 3 sub perintah, yaitu DDL (*Data Definition Language*), DML (*Data Manipulation Language*), dan DCL (*Data Control Language*). Berikut adalah pemaparan dari setiap kategori:

1. DDL (*Data Definition Language*)

   Data Definition Language yang kalau di singkat DDL merupakan perintah SQL yang berhubungan dengan pendefinisian suatu struktur *database*, dalam hal ini *database* dan *table*. Beberapa perintah dasar yang termasuk DDL ini antara lain:

   a. CREATE berfungsi untuk membuat *database* baru, tabel baru, *view* baru dan kolom.
   b. ALTER berfungsi untuk mengubah struktur tabel. Seperti mengganti nama tabel, menambah kolom, mengubah kolom, menghapus kolom maupun memberikan atribut pada kolom.
   c. DROP berfungsi untuk menghapus *database* dan tabel.
   d. TRUNCATE berfungsi untuk menghapus semua catatan dari tabel.

e. COMMENT berfungsi untuk menambahkan komentar pada data.
f. RENAME berfungsi untuk mengubah nama objek.

2. DML (*Data Manipulation Language*)

Kategori selanjutnya adalah *Data Manipulation Language* (DML). DML ialah perintah yang digunakan untuk mengelola/memanipulasi data dalam *database*. Terdapat beberapa perintah DML pada MySQL sebagai berikut:

a. SELECT berfungsi untuk mengambil/menampilkan data dari *database*.
b. INSERT berfungsi untuk memasukkan data ke dalam tabel.
c. UPDATE berfungsi untuk memperbarui data dalam tabel.
d. DELETE berfungsi untuk menghapus data dari tabel.
e. CALL berfungsi untuk memanggil sub program PL / SQL atau Java.
f. EXPLAIN PLAN berfungsi untuk menjelaskan jalur akses ke data.
g. LOCK TABLE berfungsi untuk mengunci tabel.

3. DCL (*Data Control Language*)

Kategori yang terakhir adalah *Data Control Language* (DCL). DCL merupakan perintah yang digunakan untuk melakukan pengontrolan data dan server *database*nya. Terdapat beberapa perintah DCL pada MySQL sebagai berikut:

a. GRANT berfungsi untuk memberikan hak akses pengguna ke *database*.
b. REVOKE berfungsi untuk menghilangkan hak akses yang telah diberikan dengan perintah GRANT.

2.1.13 Web Services

Web Service adalah layanan yang tersedia di Internet. Web Service menggunakan format standar XML untuk pengiriman pesannya. Web Services juga

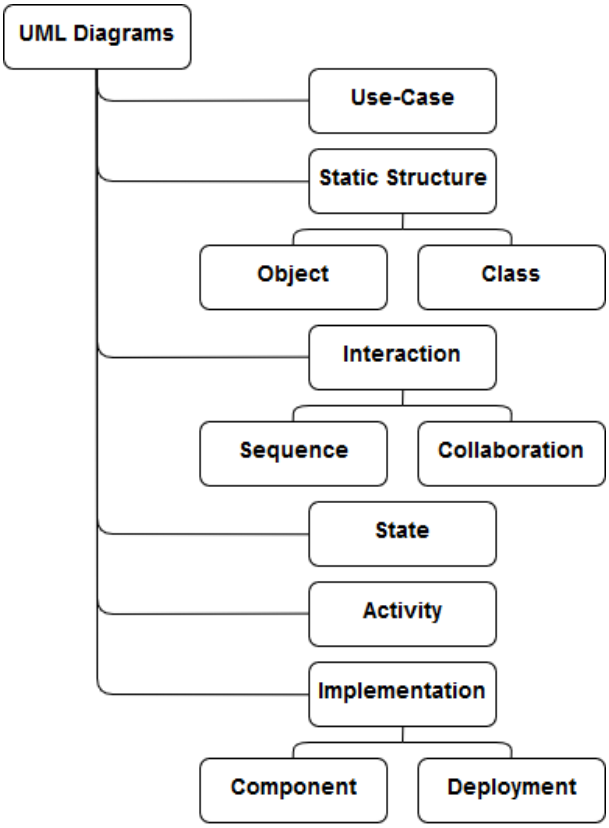
tidak terikat kepada bahasa pemrograman atau sistem operasi tertentu [24]. Web service juga memungkinkan untuk dipanggil dengan menggunakan protokol lain seperti SMTP (*Simple Mail Transfer Protocol*), namun yang paling umum digunakan HTTP. Web service dapat di definisikan sebagai aplikasi yang diakses oleh aplikasi lain.

Web service merupakan sistem *software* yang dirancang untuk mendukung *interopabilitas* mesin-ke-mesin yang dapat saling berinteraksi melalui jaringan. Web service memiliki antarmuka yang dijelaskan dalam format mesin-*processable* (khusus WSDL). Sistem lain berinteraksi dengan web service dalam cara ditentukan oleh deskripsi dengan menggunakan pesan SOAP, biasanya disampaikan menggunakan HTP dengan serial isasi XML dalam hubungannya dengan web lainnya yang terkait standar.

2.1.14 UML (*Unified Modeling Language*)

Unified Modeling Language (UML) adalah bahasa pemodelan visual yang digunakan untuk menypesifikasikan, memvisualisasikan, membangun, dan mendokumentasikan rancangan dari suatu sistem perangkat lunak [25].

Pemodelan memberikan gambaran yang jelas mengenai sistem yang akan dibangun baik dari sisi struktural ataupun fungsional. UML dapat diterapkan pada semua model pengembangan, tingkatan siklus sistem, dan berbagai macam domain aplikasi. Dalam UML terdapat konsep semantik, notasi, dan panduan masing-masing diagram. UML bertujuan menyatukan teknik-teknik pemodelan berorientasi objek-objek menjadi ter standarisasi [25]. Diagram UML dapat digambarkan sebagai berikut:



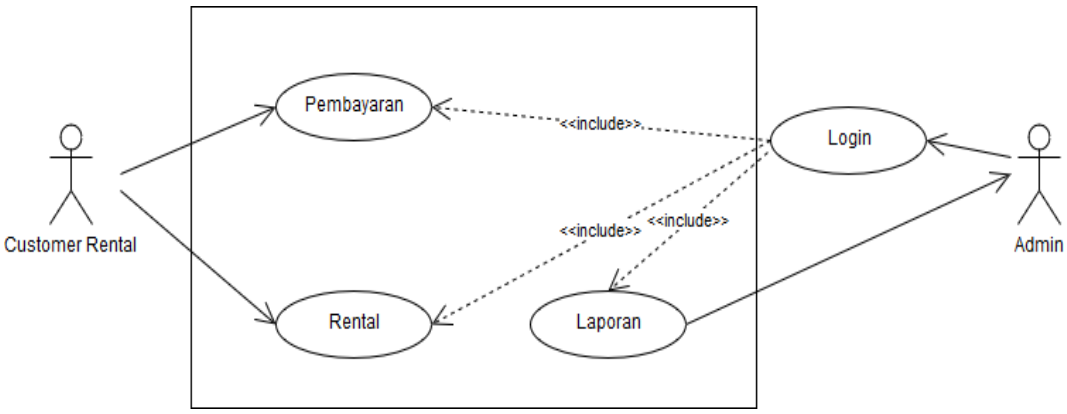
Sumber Gambar : JT-IBSI [26].

Gambar 0.4 Diagram UML

Untuk mendapatkan banyak pandangan terhadap sistem yang akan dibangun, UML menyediakan beberapa diagram visual yang menunjukkan berbagai aspek dalam sistem. Ada beberapa diagram yang disediakan dalam UML antara lain:

a. *Use case diagram*

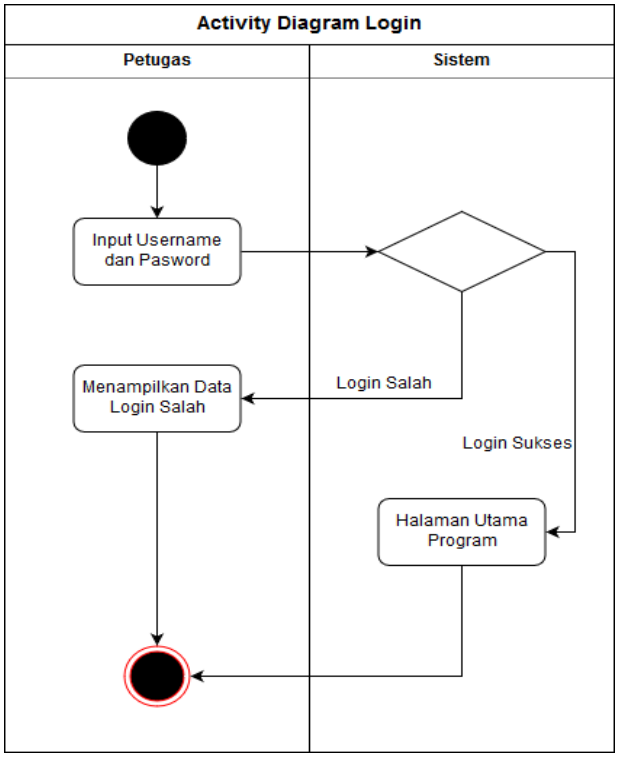
Use case diagram menyajikan interaksi antara *use case* dan aktor. Dimana, aktor dapat berupa orang, peralatan, atau sistem lain yang berinteraksi dengan sistem yang sedang dibangun. *Use case* menggambarkan fungsionalitas sistem atau persyaratan-persyaratan yang harus dipenuhi sistem dari pandangan pemakai.



Gambar 0.5 Contoh *Use Case Diagram*

b. Activity Diagram

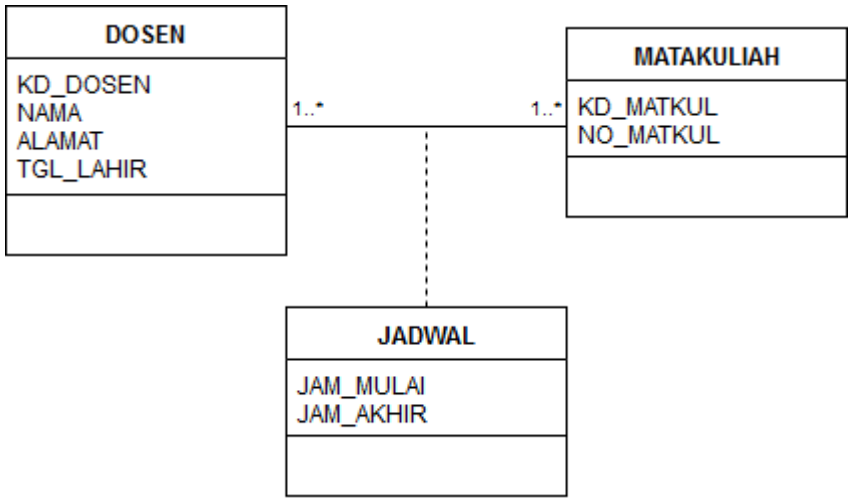
Activity diagram merupakan diagram yang menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. *Activity diagram* juga digunakan untuk mendefinisikan urutan atau pengelompokan tampilan dari sistem/*user interface* dimana setiap aktivitas dianggap memiliki sebuah rancangan antar muka tampilan serta rancang menu yang ditampilkan pada perangkat lunak.



Gambar 0.6 Contoh *Activity Diagram*

c. Class Diagram

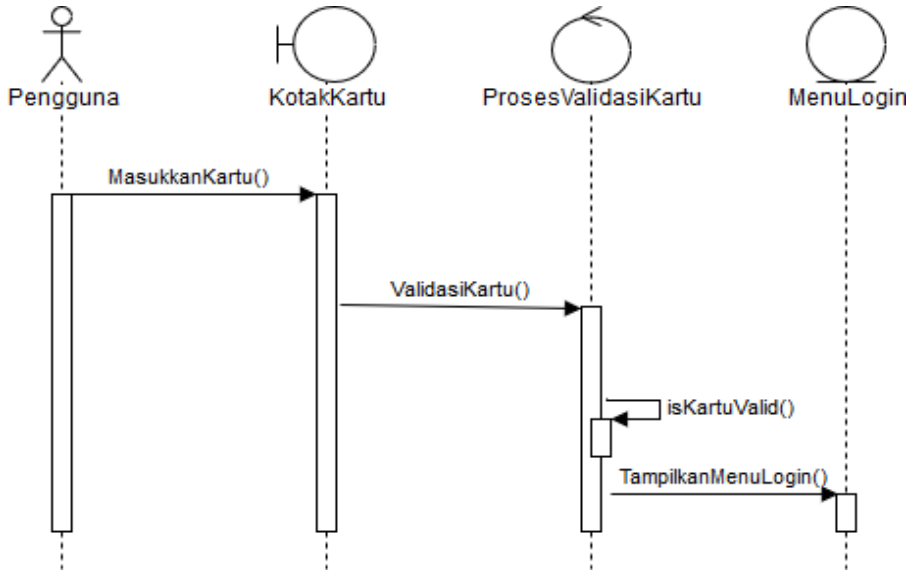
Class diagram menggambarkan suatu struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. *Class diagram* memiliki apa yang disebut atribut dan metode atau operasi. *Class diagram* mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan statis yang terdapat di antara mereka. *Class Diagram* juga menunjukkan properti dan operasi sebuah kelas dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut.



Gambar 0.7 Contoh *Class Diagram*

d. Sequence Diagram

Sequence diagram merupakan salah satu diagram *interaction* yang menjelaskan bagaimana suatu operasi itu dilakukan, *message* (pesan) apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu dan objek-objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut.



Gambar 0.8 Contoh *Sequence Diagram*

2.1.15 Webhook

Webhook adalah konsep API yang saat ini semakin populer digunakan. Semakin banyak yang kita lakukan di web, menjadikan webhook makin banyak digunakan [27]. Selain itu, webhook sangat berguna dan mudah untuk diterapkan.Webhook atau yang biasa disebut *callback* adalah cara bagi suatu aplikasi untuk menyediakan aplikasi lain dengan informasi *real-time*. Lebih mudahnya, webhook adalah *link* URL yang ditambahkan agar data yang dikirim dapat langsung diterima di waktu sama dengan *link* URL yang sudah ditentukan. Webhook merupakan satu cara yang efisien bagi *provider* maupun bagi konsumen. Satu-satunya kelemahan webhook adalah kesulitan di awal saat mengaturnya.

Webhook juga disebut web *callback*, HTTP *push* API, atau *reverse* API. Metode ini akan melakukan *callback* secara *real-time* dari aplikasi, ke server webhooks yang sudah dibuat. Server webhooks diberikan *script* untuk menjalankan beberapa perintah yang akan diproses nantinya [27].

Webhook juga biasa disebut sebagai "*Reverse* API" karena diharuskan merancang API agar webhook dapat digunakan. Langkah pertama dalam menggunakan webhook adalah memberikan URL kepada *provider* webhook untuk mengirimkan permintaan. Ini paling sering dilakukan melalui panel *backend* atau API. Juga perlu mengatur URL di aplikasi yang dapat diakses dari web publik.

Mayoritas webhook akan mengirimkan data kepada Anda dalam satu dari dua cara: sebagai JSON (biasanya) atau XML (blech) untuk kemudian ditafsirkan, atau sebagai data formulir (aplikasi / x-www-form-urlencoded atau *multipart*/*form*-data). *Provider* Anda akan memberi tahu bagaimana mereka mengirimkannya. Keduanya cukup mudah untuk dijelaskan, dan sebagian besar kerangka kerja web akan melakukan pekerjaan untuk Anda. Jika tidak, Anda mungkin perlu menggunakan satu atau dua fungsi lain.

Dengan Webhook aplikasi dapat menerima pemberitahuan ketika terdapat perubahan terhadap serangkaian topik yang dipilih beserta kolomnya. Misalnya, mengatur Webhook untuk topik *User* dan berlangganan ke kolom email, serta akan diberi tahu ketika pengguna memperbarui alamat emailn ya. Hal ini mencegah agar tidak bergantung pada permintaan API berkelanjutan atau bahkan periodik untuk memeriksa pembaruan yang mungkin atau tidak mungkin terjadi. Hal ini juga membantu untuk menghindari pembatasan laju [3].

Dalam metode Webhook, *client* dapat selalu bisa mengakses bot kapan pun tanpa harus desktop yang digunakan untuk membuat dan merancang bot menyala asalkan server yang digunakan tetap dalam kondisi menyala. Berbeda dengan metode *Long Polling* yang mengharuskan desktop yang digunakan untuk membuat dan merancang bot menyala [6].

Sumber Gambar : Jurnal ISSN [28].

Gambar 0.9 Cara Kerja Webhook

2.1.16 Google Event Calendar

Google Calendar adalah layanan kalender yang disediakan Google secara gratis bagi pelanggannya. Pemilik akun Google dapat membuat kalender, membuat acara dan mengundang orang lain ke dalam acara tersebut. Google Calendar dapat diakses melalui telepon genggam dan dapat memberi peringatan melalui SMS atau Surel. Google menyediakan *Application* API yang mengizinkan pengembang untuk membuat aplikasi yang dapat mengakses aplikasi-aplikasi Google seperti Gmail, Calendar dan aplikasi lain menggunakan Google Data APIs, Gadgets, dan Google Apps Script.

Google Calendar yaitu agenda *online* dengan fasilitas berbaginya, dan Google Talk yang digunakan untuk mengobrol secara *online*. Kedua, perangkat kolaborasi yang terdiri dari Google Sites, Google Docs dan *Address Book*. Terakhir, perangkat keamanan yang dikenal dengan Postini Services. Semua aplikasi dan service Google Apps dikaitkan dengan sebuah domain.

Google Calendar merupakan salah satu produk layanan berbasis kalender dari google yang dapat diakses secara *online* dan menyediakan berbagai macam fitur yang bisa digunakan secara gratis. Fitur-fitur yang dapat digunakan pada google calendar adalah membuat dan mengatur jadwal, berbagi kalender, pengingat jadwal acara dengan email, *popup*, dan sms dan dapat mengimpor jadwal acara ke

kalender lain. Untuk pengembangannya google calendar membutuhkan google calendar API yang digunakan untuk membangun sebuah aplikasi *client*. Google Calendar API mendukung beberapa macam bahasa pemrograman untuk pengembangan aplikasi berbasis kalender, bahasa pemrograman tersebut antara lain PHP, Java, .NET, Python dan Ruby (*Overview*, Google Developers) [7].

Penggunaan Google Event Calendar yang tidak terbatas hanya pada satu orang saja membuatnya lebih efisien. Seluruh pengguna sistem yang sudah dicantumkan pada event kalendar tersebut dapat menerima notifikasi *reminder* tersebut. Google Calendar juga mampu mengingatkan secara kelompok atau grup. Google Calendar merupakan salah satu aplikasi berupa kalender digital yang dapat dimanfaatkan untuk membuat jadwal dan pengingat pengguna [8].

Kelebihan dari aplikasi kalender ini adalah integrasi dengan Google Calendar. Semua ruangan direpresentasikan sebagai Google Calendar Resource yang masing-masing *resource* memiliki sebuah *calendar* sendiri dan masing-masing *calendar* memiliki *event*. Tiap event merepresentasikan suatu kegiatan, misalnya: jadwal pengerjaan proyek tukang. Data Google Calendar dapat diakses menggunakan aplikasi kalender dalam telepon cerdas dan Google juga dapat memberi notifikasi melalui email dan SMS.

Pada bagian ini dijelaskan keterhubungan kalender dengan *event*, kalender adalah kumpulan acara terkait, bersama dengan meta data tambahan seperti ringkasan, zona waktu *default*, lokasi, dll. Setiap kalender diidentifikasi oleh ID yang merupakan alamat email. Kalender dapat memiliki banyak pemilik.

Suatu *event* adalah objek yang terkait dengan tanggal atau rentang waktu tertentu. Acara diidentifikasi oleh ID yang unik dalam kalender. Selain waktu tanggal mulai dan berakhir, acara berisi data lain seperti ringkasan, deskripsi, lokasi, status, pengingat, lampiran, dll.

Jenis *event*,

Kalender Google mendukung *event* tunggal dan berulang:

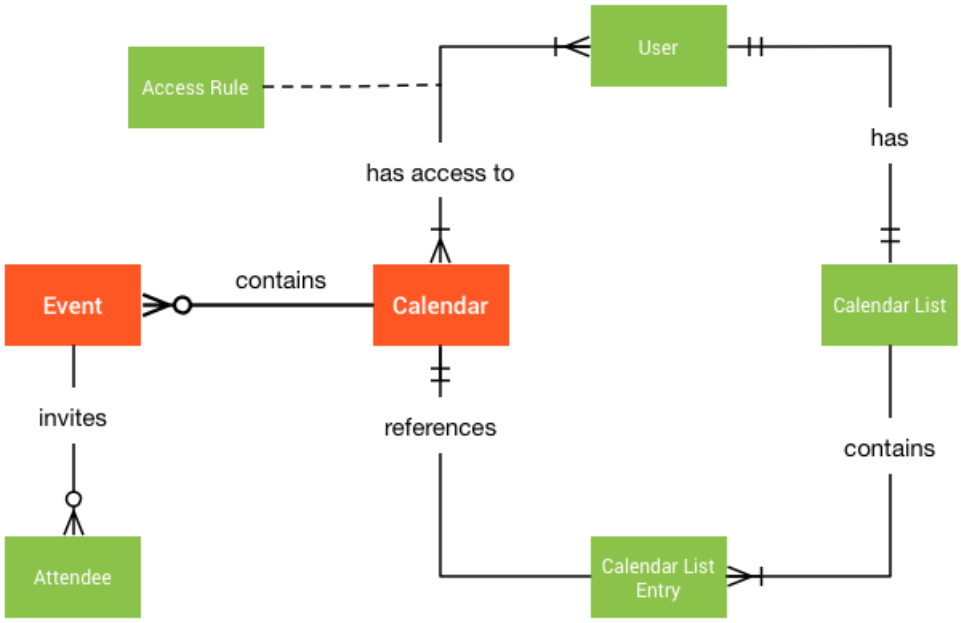
1. Satu peristiwa mewakili kejadian unik.
2. Peristiwa berulang mendefinisikan beberapa kejadian.

Event juga dapat diatur waktunya atau sepanjang hari:

1. Event berjangka waktu terjadi antara dua titik tertentu dalam waktu. Event berjangka waktu menggunakan bidang start.dateTime dan end.dateTime untuk menentukan kapan *event* itu terjadi.
2. Event sepanjang hari berlangsung sepanjang hari atau serangkaian hari berturut-turut. Event sepanjang hari menggunakan bidang *start.date* dan *end.date* untuk menentukan kapan *event* itu terjadi. Perhatikan bahwa bidang zona waktu tidak memiliki arti penting untuk *event* sepanjang hari.

Events memiliki satu penyelenggara yang merupakan kalender yang berisi salinan utama *events*. Acara juga dapat memiliki banyak peserta. Peserta biasanya adalah kalender utama dari pengguna yang diundang.`

Gambar berikut menunjukkan hubungan konseptual antara kalender, *events*, dan elemen terkait lainnya:



Sumber : https://developers.google.com/calendar/concepts/events-calendars

Gambar 0.10 Konsep Google Event Calendar

2.1.17 Sistem Rekomendasi

Sistem rekomendasi adalah suatu program yang melakukan prediksi sesuatu item, seperti rekomendasi film, musik, buku, berita dan lain sebagainya yang menarik *user*. Sistem ini berjalan dengan mengumpulkan data dari *user* secara langsung maupun tidak [29].

Pengumpulan data secara langsung dapat dilakukan sebagai berikut:

1. Meminta *user* untuk melakukan *rating* pada sebuah item.
2. Meminta *user* untuk melakukan rangking pada item favorit setidaknya memilih satu item favorit.
3. Memberikan beberapa pilihan item pada *user* dan memintanya memilih yang terbaik.
4. Meminta *user* untuk mendaftar item yang paling disukai atau item yang tidak disukainya.

Pengumpulan data dengan tidak langsung berhubungan dengan seorang user, dilakukan dengan cara sebagai berikut:

1. Mengamati item yang dilihat oleh seorang *user* pada sebuah web *e-commerce*.
2. Mengumpulkan data transaksi pada sebuah toko *online*.

Data hasil pengumpulan, kemudian dilakukan perhitungan dengan algoritma tertentu yang kemudian hasil tersebut dikembalikan lagi kepada *user* sebagai sebuah rekomendasi item dengan parameter dari *user* tersebut. Sistem rekomendasi juga merupakan salah satu alternatif sebagai mesin pencari suatu item yang dicari oleh *user*. *Collaborative Filtering* merupakan salah satu item *based* dari sistem rekomendasi.

2.1.18 *Collaborative Filtering*

Collaborative filtering merupakan salah satu algoritma yang digunakan untuk menyusun *recommender system* dan telah terbukti memberikan hasil yang sangat baik. *Rating* produk merupakan elemen terpenting dari algoritma ini, *rating* diperoleh dari sebagian besar *customer* di mana *customer* secara *explicit* memberikan penilaiannya terhadap produk. Kesimpulannya ialah *system*

memberikan timbal balik kepada *customer* dengan mengolah data-data tersebut, sebagai gambaran dari skala nol sampai 5 yang mengindikasikan penilaian yang paling tidak disukai hingga paling disukai menurut sudut pandang *customer*, data ini memungkinkan untuk dilakukannya perhitungan statistik yang hasilnya menunjukkan produk mana yang diberikan *rating* tinggi oleh *customer* [30].

Collaborative filtering menggunakan *database* yang diperoleh dari *user*. Ada dua komponen utama dalam data ini agar dapat membuat prediksi bagi *recommender system* yaitu *user* dan item. Keduanya membentuk *rating matrix* berupa m *user* {u1, u2, u3, …, um} dan daftar n item {i1, i2, i3, …, i4}. Di mana setiap *user* memberikan penilaiannya pada item berupa *rating* dalam skala 1 sampai 5. *Rating* ini dilambangkan dengan Iu1. Tidak semua *user* memberikan *rating* ke setiap produk karena berbagai macam faktor, hal ini menyebabkan banyaknya *missing value* yang mengakibatkan *sparsity* pada data. *User* item *rating matrix* dapat digambarkan dengan *table* di bawah.

Tabel 0.1 Rating Matrix

	$\mathbf{i_1}$	$\mathbf{i_2}$	$\mathbf{i_3}$	$\mathbf{i_4}$	$\mathbf{i_m}$
u	1	...	3	...	
$\mathbf{u_2}$	5	4	...	...	
$\mathbf{u_3}$	...	5	3	...	
$\mathbf{u_4}$	...	4	...	...	
$\mathbf{u_m}$					

Sumber Tabel : Jurnal Ilmiah Teknologi Informasi Terapan [30]**.**

Collaborative filtering dibagi menjadi dua kategori yaitu *memory based*, *model based* dan gabungan keduanya menjadi *hybrid recommendation system* bertujuan untuk mengatasi kelemahan yang muncul pada kedua kategori sebelumnya. *Memory based recommendation* menggunakan *user rating* sebagai bahan untuk menemukan *similarity* atau derajat kesamaan antar *user*. Di domain bisnis algoritma ini telah diterapkan pada situs *Amazon*, keunggulannya adalah kemudahan dalam implementasi dan sangat efektif. Sedangkan untuk model *based recommendation* tidak jauh berbeda dengan algoritma sebelumnya, yaitu menggunakan *rating* sebagai sumber data. Namun algoritma ini menggunakan teknik-teknik di data

mining atau *machine learning* seperti *Bayesian* dan *clustering*. Gabungan dari model dan *memory based* membentuk *hybrid recomender system*. Algoritma ini di ciptakan untuk mengatasi kelemahan yang terdapat pada dua algoritma sebelumnya seperti *sparsity*. Berikut adalah perbandingan antara *memory based* dan model *based recommendation system* [30]:

1. *Memory Based*

Teknik yang dipakai adalah: *Neighbor-based* CF (*item based/user-based* CF. *Algorithms with Pearson/vector cosine correlation*) *Item-based/user-based* top-N *Recommendations*. Keunggulannya berupa implementasi mudah, mudah menambahkan data-data baru tidak perlu mempertimbangkan *content item* yang direkomendasikan, skala yang baik dengan *co-rated item*. Sedangkan kekurangan dari algoritma ini adalah bergantung pada *rating* dari *user*, menurunnya performa jika data jarang, skalabilitas yang terbatas pada data set yang besar.

2. *Model Based*

Teknik yang dipakai adalah: Bayesian belief nets CF, clustering CF, MDP-based CF, *latent semantic* CF, *sparse factor analysis*, CF *using dimensionality*, *reduction techniques* seperti SVD dan PC. Kelebihan: dapat mengatasi masalah data yang jarang, skalability dan masalah lainnya, akurasi meningkat, memberikan *intuitive rationa*l untuk rekomendasi. Kekurangannya adalah diperlukannya sumber daya yang besar untuk proses komputasi.

3. *Hybryd Recommendation System*

Teknik yang digunakan berupa *content based* CF (fab), content boosted CF, *hybrid* CF kombinasi *memory based* dan *model based* (*personality* diagnosis). Teknik ini sebagai solusi atas kelemahan yang terdapat pada *memory* dan *model based* CF seperti *sparsity* dan *grayship*, sehingga meningkatkan akurasi prediksi. Beberapa kelemahan juga muncul seperti meningkatnya kompleksitas program hingga perlunya data *external* yang terkadang tidak tersedia.

2.1.19 *Slope One*

Algoritma *Slope One* adalah bentuk yang paling mudah dari teknik *Collaborative Filtering* barang yang berbasis pada *rating*. Kemudahan ini

menyebabkan algoritma ini mudah untuk diterapkan dengan tingkat ketepatan yang tidak kalah dari algoritma dengan perhitungan yang jauh lebih sulit. Dan algoritma ini akhirnya digunakan sebagai dasar untuk pengembangan beberapa algoritma lain.

Terdapat beberapa metode yang dapat digunakan untuk membuat sebuah sistem rekomendasi diantaranya *Collaborative filtering*, *Content-based filtering*, dan *Hybrid*. Algoritma *Slope One* termasuk dalam *Item-based Collaborative filtering* yang merupakan pengembangan dari metode *Collaborative filtering*. Perbedaan *Item-based Collaborative filtering* dengan *Collaborative filtering* adalah *pada Item-based,* hasil *rating* yang dibandingkan untuk memberikan rekomendasi.

Algoritma *slope one* berjalan dengan menghitung selisih *rating* antara dua item yang ada. Selisih *rating* tersebut digunakan untuk memprediksi berapa besar nilai *rating* yang diberikan terhadap sebuah item untuk pelanggan P. Jadi ada dua masukan untuk algoritma *slope one* yaitu *rating* dari pelanggan P dan item mana yang akan diprediksi.

Algoritma *Slope One* adalah salah satu algoritma untuk membuat sistem rekomendasi. *Slope one* memberikan prediksi berdasarkan nilai hasil pencarian dari item–item yang dibandingkan. Keunggulan algoritma *Slope One* dibandingkan algoritma rekomendasi lainnya adalah algoritma *Slope One* mudah untuk diimplementasi, efisien saat melakukan *query*, tidak memerlukan banyak *requirement* dikarenakan rekomendasi berdasarkan *rating* dari setiap item, dan cukup akurat [5].

Algoritma *Slope One* melakukan perhitungan berdasarkan hubungan linear dari nilai preferensi atau *weight* dari setiap item yang dibandingkan. Estimasi umum dari dasar perhitungan algoritma *Slope One* adalah fungsi linear y = mx + b, dengan asumsi *gradient* m = 1, sehingga fungsi menjadi b = y -x. Cara kerja algoritma *Slope One* adalah dengan mencari selisih dari suatu item dengan item-item lain yang dibandingkan.

Perhitungan algoritma *Slope One* dapat diformulasikan dengan persamaan (1) untuk pencarian selisih [5].

$$dev_{j,i} = \sum u \in S_{j,i}(x) \frac{uj - ui}{card(S_{j,i}(x))} \quad .....(2.1)$$

Dimana:

$dev_{j,i}$ = rata-rata selisih rating item j dan i.

u_j = rating item j.

u_i = rating item i

card ($S_{j,i}(x)$) = banyaknya elemen yang dibandingkan.

Apabila selisih sudah didapatkan, maka dapat dilakukan perhitungan rekomendasi untuk item j yang dapat dirumuskan dengan persamaan (2):

$$p^{SI}(u)_j = dev_{j,i} + u_j \quad .....(2.2)$$

Dimana:

$p^{SI}(u)_j$ = nilai rekomendasi untuk item j.

Berdasarkan persamaan di atas, algoritma *Slope One* memberikan rekomendasi dengan melakukan perhitungan selisih dari setiap item. Selisih yang didapat akan dirata-rata per item yang kemudian akan dijumlahkan dengan *value* dari masing-masing item. *Value* yang sudah dijumlahkan dengan rata-rata selisihnya akan digunakan sebagai *point* untuk memberikan rekomendasi [5]