BAB 2
TINJAUAN PUSTAKA

2.1 Landasan Teori

Landasan teori merupakan penjelasan dasar teori-teori yang berkaitan dalam pembangunan aplikasi pemesanan dan pemesangan *wallpaper* dengan teknologi arcore berbasis android. Teori-teori yang terkait serta mendukung penelitian ini akan dibahas pada bab ini.

2.2 *Wallpaper* Dinding

Wallpaper dinding merupakan hiasan dinding berupa gulungan kertas yang mempunyai berbagai macam motif dan warna. *Wallpaper* dinding seringkali digunakan untuk merubah suasana dan penampilan dari sebuah properti agar terlihat mewah dan mempunyai nilai tambah secara signifikan. Selain pengerjaan yang sangat cepat, pengaplikasian *wallpaper* dinding juga tidak menimbulkan bau apapun mungkin karena faktor itu banyak orang yang mulai menggunakan *wallpaper* dinding dibandingkan mengecat temboknya, banyak sekali yang menggunakan *wallpaper* dinding untuk pengisian ruko, rumah, kantor, hotel dan juga yang hendak melakukan renovasi pada propertynya, karena cenderung lebih irit dan terlihat lebih baik dengan maintenance yang relatif sangat mudah.

Gambar 2. 1 Contoh Motif *Wallpaper* [3]

pada beberapa jenis *wallpaper* dinding jika dilihat dari bahannya, yaitu :

1. *Vinyl Backing wallpaper*

   Vinyl backing wallpaper merupakan *wallpaper* dinding yang terbuat dari bahan vinyl .*Wallpaper* yang terbuat dari bahan vinyl ini adalah jenis *Wallpaper* Dinding yang paling banyak kita temukan di Negara Tercinta ini di Indonesia. Selain harganya yang terjangkau, *Vinyl backing wallpaper* ini juga memiliki ketahanan yang relatif lebih lama.
2. *Non – Woven backing wallpaper*

   Non Woven Backing Wallpaper adalah *Wallpaper* Dinding yang terbuat dari bahan *non woven*. *Non woven wallpaper* jarang ditemukan di Indonesia, karena harganya yang terlalu tinggi dan pemasangan nya yang relatif lebih rumit dan kompleks dibandingkan dengan *vinyl wallpaper*. Keunggulan dari *Non woven wallpaper* adalah *wallpaper* dinding ini dapat dipindah tempatkan setelah lama terpasang.

3. *Pure Backing wallpaper*

   Pure backing wallpaper adalah *Wallpaper* Dinding yang terbuat dari bahan dasar kertas. Wallpaper yang terbuat dari bahan kertas ini tidak terlalu favorit dan tidak dianjurkan untuk dipasang karena terlalu gampang sobek dan lecet. *Wallpaper* jenis ini relatif terlalu tipis, sehingga rentan sobek pada saat pemasangan dilakukan.

4. *Fabric Backing Wallpaper*

   Fabric Backing Wallpaper adalah *Wallpaper* yang terbuat dari bahan jenis *fabric*. *Wallpaper* jenis ini jarang sekali ditemukan, dan kebanyakan *made by order*. Harga dari *Fabric Backing Wallpaper* sangatlah tinggi. Karena terbuat dari bahan rajutan yang biasanya sangat tebal. Pengaplikasian pada saat pemasangan juga lebih rumit dan berat. Lem *wallpaper* yang digunakan pun haruslah lem *wallpaper* khusus untuk menopang *Fabric Wallpaper* ini. *Wallpaper* jenis ini banyak digunakan di Istana Kerajaan pada jaman dahulu [3].

2.3 Sistem

Sistem adalah suatu kesatuan yang terdiri dari komponen atau elemen yang dihubungkan bersama untuk memudahkan aliran informasi, materi, atau energi. Menurut Ludwig Von Bartalanfy, merupaan seperangkat unsur yang saling terkait dalam suatu antar relasi di antara unsur-unsur tersebut dengan lingkungan. Menurut Anatol Raporot, sistem adalah suatu kumpulan kesatuan dan perangkat hubungan satu sama lain. Menurut L. Ackof, Sistem adalah setiap kesatuan secara konseptual atau fisik yang terdiri dari bagian-bagan dalam keadaan saling tergantung sama lainnya.

2.3.1 Karakteristik Sistem

Menurut Kusrini dan Koniyo (2007:6) sistem mempunyai karakteristik atau sifat-sifat tertentu, diantaranya:

1. Komponen Sistem (*Component*)

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang saling bekerja sama membentuk suatu komponen sistem atau bagian-bagian dari sistem.

2. Batas Sistem (*Boundary*)

   Merupakan daerah yang membatasi suatu sistem dengan sistem yang lain atau dengan lingkungan kerjanya.

3. Subsistem

   Bagian-bagian dari sistem yang beraktivitas dan berinteraksi satu sama lain untuk mencapai tujuan dengan sasarannya masing-masing.

4. Lingkungan Luar Sistem (*Environment*)

   Suatu sistem yang ada diluar dari atas sistem yang dipengaruhi oleh operasi sistem.

5. Penghubung Sistem (*Interface*)

   Media penghubung antara suatu subsistem dengan subsistem lain. Adanya penghubung ini memungkinkan berbagai sumber daya mengalir dari suatu subsistem ke subsistem lainnya.

6. Masukan Sistem (*Input*)

   Energi yang masuk ke dalam sistem, berupa perawatan dan sinyal. Masukkan perawatan adalah energi yang dimasukkan supaya sistem tersebut dapat berinteraksi.

7. Keluaran Sistem (*Output*)

   Hasil energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

8. Pengolahan Sistem (*Proces*)

   Suatu sistem yang mempunyai suatu bagian pengolah yang akan mengubah masukan menjadi keluaran.

9. Sasaran Sistem (*Objective*)

   Tujuan yang ingin dicapai oleh sistem, akan dikatakan berhasil apabila mengenai sasaran atau tujuan.

2.4 Rekomendasi

Sistem rekomendasi merupakan model aplikasi dari hasil observasi terhadap keadaan dan keinginan pengguna. Sistem rekomendasi memerlukan model rekomendasi yang tepat agar apa yang direkomendasikan sesuai ddengan keinginan pengguna, serta mempermudah pengguna mengambil keputusan yang tepat dalam menentukan item atau tempat mana yang akan dipilih [4].

Sistem rekomendasi memberikan saran menurut selera atau ukuran tertentu tentang item yang mungkin menarik minat pengguna dengan cara mencocokkan pilihan dengan profil pengguna atau grup pengguna. Sistem ini harus cerdas dan bijaksana memahami apa yang pengguna inginkan. Dengan kata lain, system cukup bijak untuk mempelajari penggunanya, membuat profil mereka , dan memberikan rekomendasi atau saran.

2.5 Metode *Collaborative Filtering*

Collaborative filtering merupakan salah satu algoritma yang digunakan untuk menyusun sistem rekomendasi 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 penilaiaannya terhadap produk. Kesimpulannya ialah sistem memberikan timbal balik kepada customer dengan mengolah data-data tersebut, sebagai gambaran dari skala nol sampai lima 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* [5].

Collaborative filtering menggunakan database yang diperoleh dari *user*. Ada dua komponen utama dalam data ini agar dapat membuat prediksi bagi *recommender* sistem 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 tabel di bawah.

Tabel 2. 1 Rating Matrix

	$\mathbf{i_1}$	$\mathbf{i_2}$	$\mathbf{i_3}$	$\mathbf{i_4}$	$\mathbf{i_m}$
$\mathbf{u_1}$	1	...	3	...	
$\mathbf{u_2}$	5	4	...	...	
$\mathbf{u_3}$	...	5	3	...	
$\mathbf{u_4}$	...	4	...	...	
$\mathbf{u_m}$					

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 recomennder 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* [5]:

a. 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 menpertimbangkan content item yang direkomendasikan, skala yang baik dengan co-rated item. Sedangkan kekurangan dari algortima ini adalah bergantung pada rating dari user, menurun-nya performa jika data jarang, skalabilitas yang terbatas pada dataset yang besar.

b. 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 rational untuk recomendasi. Kekurangannya adalah diperlukannya sumber daya yang besar untuk proses komputasi.

c. 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 eksternal yang terkadang tidak tersedia.

2.6 Aplikasi

Aplikasi adalah Program siap pakai yang dapat digunakan untuk menjalankan perintah-perintah dari pengguna aplikasi tersebut dengan tujuan mendapatkan hasil yang lebih akurat sesuai dengan tujuan pembuatan Aplikasi tersebut, aplikasi mempunyai arti yaitu pemecahan masalah yang menggunakan salah satu teknik pemrosesan data aplikasi yang biasanya berpacu pada sebuah komputansi yang diinginkan atau diharapkan maupun pemrosesan data yang diharapkan.

Pengertian Aplikasi secara umum adalah alat terapan yang difungsikan secara khusus dan terpadu sesuai kemampuan yang dimilikinya, Aplikasi merupakan suatu perangkat komputer yang siap pakai bagi user. Pengertian Aplikasi menurut para ahli:

1. Pengertian Aplikasi menurut Jogiyanto (1999:12) adalah penggunaan dalam suatu komputer, intruksi (*instructiom*) atau pernyataan (*statement*) yang disusun sedemikian rupa sehingga computer dapat memproses *input* menjadi *output*.
2. Pengertian Aplikasi menurut Kamus Besar Bahasa Indonesia (1998:52) adalah penerapan dari rancang sistem untuk mengolah data yang menggunakan aturan atau ketentuan bahasa pemrograman tertentu. Aplikasi adalah suatu program computer yang dibuat untuk mengerjakan dan melaksanakan tugas khusus dari pengguna.
3. Menurut Rachmad Hakim S.,Aplikasi adalah perangkat lunak yang digunakan untuk tujuan tertentu,seperti mengolah dokumen, mengatur Windows & permainan (game), dan sebagainya.
4. Menurut Harip Santoso, Aplikasi adalah suatu kelompok *file* (*form, class, report*) yang bertujuan untuk melakukan aktivitas tertentu yang saling terkait, misalnya aplikasi *payroll*, aplikasi *fixed asset*, dll [6].

2.7 ARCORE

ARCore merupakan pengembangan dari Project Tango, sebagaimana Google ingin agar pengalaman Augmented Reality ini bisa dirasakan oleh banyak orang para pengguna ponsel Android maka diciptakanlah ARCore ini oleh Google. Keduanya memang memberikan pengalaman dalam hal AR, namun pada Project Tango diperlukan spesifikasi hardware yang lebih, dibutuhkan dua kamera untuk menjalankan aplikasi AR seperti yang ada pada ponsel Asus ZenFone AR.

Di Project Tango, pengguna perlu berinvestasi membeli ponsel dengan spesifikasi semacam Asus dan dari manufaktur pembuat ponsel Android pun perlu membuat ponsel berspesifikasi serupa. Berbeda dengan ARCore, siapapun pengguna Android bisa menikmati pengalaman AR dengan semua ponsel asal sudah memiliki versi Android Nougat atau di atasnya. Saat ini Tango memberikan pengalam AR yang lebih real karena dukungan hardware dibandingkan dengan ARCore. Walau begitu, seiring berjalannya waktu, Google akan meminta dan menekan para pembuat ponsel untuk meningkatkan kemampuan kamera maupun spesifikasi lainnya untuk mendukung perkembangan ARCore [7].

ARCore menggunakan tiga kemampuan dalam mengintegrasikan konten virtual ke dalam dunia nyata seperti yang nantinya akan terlihat dalam kamera ponsel. Tiga kemampuan tersebut terdiri dari Motion Tracking (memungkinkan ponsel paham dan melacak posisi relatifnya terhadap dunia nyata), Enviromental understanding (memungkinkan ponsel untuk mendeteksi ukuran dan lokasi semua tipe permukaan, horisontal, vertikal dan sudut), dan Light Estimation (memungkinkan ponsel mengestimasi kondisi pencahayaan ruangan).

Seperti dijelaskan pada halaman developer Google, secara fundamental ARCore melakukan dua hal yaitu melacak posisi ponsel saat bergerak dan membangun pemahaman sendiri terhadap kondisi real atau dunia nyata. Teknologi pelacakan gerak ini menggunakan kamera ponsel untuk mengidentifikasi titik-titik yang menarik, fitur-

fitur yang disediakan, dan melacak pergerakan titik-titik tersebut setiap waktu. Kombinasi antara pergerakan titik-titik dan pembacaan melalui sensor inersia ponsel, ARCore mampu menentukan baik posisi maupun orientasi ponsel atas pergerakannya terhadap ruang.

ARCore mampu mendeteksi permukaan datar seperti meja maupun lantai, mampu pula mengestimasi pencahayaan ruang. Kapabilitas ini bergabung untuk memampukan ARCore membangun pemahaman tersendiri tentang kondisi real di sekitarnya.

Dengan kemampuannya,pengguna bisa meletakkan objek apapun di dalamnya, anotasi atau catatan maupun informasi lain yang terintegrasi secara sempurna dengan dunia nyata. Google memberikan contoh, dengan meletakkan anak kucing di bagian pojok meja kopi, atau memberikan anotasi pada sebuah lukisan mengenai informasi biografi artis yang ada dalam lukisan tersebut. *Motion tracking* atau pelacakan gerak mengartikan bahwa kita dapat bergerak dan melihat objek dari susut tertentu, dan bahkan apabila kita berputar balik dan meninggalkan ruangan, ketika kita kembali, maka anak kucing ataupun anotasi akan tetap berada pada posisi seperti sesaat kita meninggalkannya .

Adapun hubungan ARCORE pada penelitian ini karena aplikasi yang dibuat sangat membutuhkan fitur ini agar pengguna bisa mengukur luas dinding dengan memanfaantkan augmented reality yang di sediakan oleh ARCORE.

2.8 Java

Java adalah sebuah bahasa pemrograman yang diciptakan oleh James Gosling, seorang developer dari *Sun Microsystem* pada tahun 1991. Selanjutnya Java dikembangkan Sun Microsystem dan banyak digunakan untuk menciptakan *Executable Content* yang dapat didistribusikan melalui *network.*

Java adalah bahasa pemrograman *Object-Oriented* dengan unsur-unsur seperti bahasa C++ dan bahasa-bahasa lainnya yang memiliki *libraries* yang cocok untuk lingkungan internet. Java dapat melakukan banyak hal dalam melakukan

pemrograman, seperti membuat animasi halaman *web*, pemrograman Java untuk Ponsel dan aplikasi interaktif.

2.8.1 Karakteristik Java

Java adalah sebuah bahasa pemrograman berorientasi obyek murni. Jadi program – program Java berada dalam sebuah struktur kelas – kelas dan obyek – obyek. Pada dasarnya sintaks pada bahasa Java mirip dengan sintaks pada bahasa C atau C++. Java bertipe kuat (*strongly-typed*). Ini berarti semua tipe data terikat secara statis atau dengan kata lain setiap nama variabel diasosiasikan dengan sebuah tipe data tunggal yang dikenali pada saat kompilasi.

2.8.2 Kelebihan Java

1. *Multiplatform*

Kelebihan utama dari Java ialah dapat dijalankan di beberapa platform / sistem operasi komputer, sesuai dengan prinsip tulis sekali, jalankan di mana saja. Dengan kelebihan ini pemrogram cukup menulis sebuah program Java dan dikompilasi (diubah, dari bahasa yang dimengerti manusia menjadi bahasa mesin /*bytecode*) sekali lalu hasilnya dapat dijalankan di atas beberapa platform tanpa perubahan. Kelebihan ini memungkinkan sebuah program berbasis java dikerjakan diatas operating system Linux tetapi dijalankan dengan baik di atas Microsoft Windows. Platform yang didukung sampai saat ini adalah Microsoft Windows, Linux, Mac OSdan Sun Solaris. Penyebanya adalah setiap sistem operasi menggunakan programnya sendiri-sendiri (yang dapat diunduh dari situs Java) untuk meninterpretasikanbytecode tersebut.

2. OOP (*Object Oriented Programming* - Pemrogram Berorientasi Objek)

yang artinya semua aspek yang terdapat di Java adalah Objek. Java merupakan salah satu bahasa pemrograman berbasis oebjek secara murni. Semua tipe data diturunkan dari kelas dasar yang disebut Object. Hal ini sangat memudahkan pemrogram untuk mendesain, membuat, mengembangkan dan mengalokasi kesalahan sebuah program dengan basis Java secara cepat, tepat, mudah dan terorganisir.

Kelebihan ini menjadikan Java sebagai salah satu bahasa pemograman termudah, bahkan untuk fungsi fungsi yang advance seperti komunikasi antara komputer sekalipun.

3. Perpustakaan Kelas Yang Lengkap

Java terkenal dengan kelengkapan library/perpustakaan(kumpulan program program yang disertakan dalam pemrograman java) yang sangat memudahkan dalam penggunaan oleh para pemrogram untuk membangun aplikasinya. Kelengkapan perpustakaan ini ditambah dengan keberadaan komunitas Java yang besar yang terus menerus membuat perpustakaan-perpustakaan baru untuk melingkupi seluruh kebutuhan pembangunan aplikasi.

4. Bergaya C++

memiliki sintaks seperti bahasa pemrograman [C++] sehingga menarik banyak pemrogram C++ untuk pindah ke Java. Saat ini pengguna Java sangat banyak, sebagian besar adalah pemrogram C++ yang pindah ke Java. Universitas-universitas di Amerika juga mulai berpindah dengan mengajarkan Java kepada murid-murid yang baru karena lebih mudah dipahami oleh murid dan dapat berguna juga bagi mereka yang bukan mengambil jurusan komputer.

5. Pengumpulan sampah otomatis

memiliki fasilitas pengaturan penggunaan memori sehingga para pemrogram tidak perlu melakukan pengaturan memori secara langsung (seperti halnya dalam bahasa C++ yang dipakai secara luas).

2.8.3 Kekurangan Java

1. Tulis sekali,perbaiki di mana saja - Masih ada beberapa hal yang tidak kompatibel antara platform satu denganplatform lain. Untuk J2SE, misalnya SWT-AWT bridge yang sampai sekarang tidak berfungsi pada Mac OS X.

2. Mudah didekompilasi. Dekompilasi adalah proses membalikkan dari kode jadi menjadi kode sumber. Ini dimungkinkan karena koe jadi Java merupakan *bytecode* yang menyimpan banyak atribut bahasa tingkat tinggi, seperti nama-nama kelas, metode, dan tipe data. Hal yang sama juga terjadi pada *Microsoft .NET Platform*. Dengan demikian, algoritma yang digunakan program akan lebih sulit disembunyikan dan mudah dibajak/*direverse-engineer*.

3. Penggunaan memori yang banyak. Penggunaan memori untuk program berbasis Java jauh lebih besar daripada bahasa tingkat tinggi generasi sebelumnya seperti C/C++ dan Pascal (lebih spesifik lagi, Delphi dan Object Pascal). Biasanya ini bukan merupakan masalah bagi pihak yang menggunakan teknologi terbaru (karena trend memori terpasang makin murah), tetapi menjadi masalah bagi mereka yang masih harus berkutat dengan mesin komputer berumur lebih dari 4 tahun.

Dalam penelitian ini menggunakan bahasa pemrograman java dikarenakan menggunakan tools Android Studio, tools tersebut mempunyai basic java. Maka dari itu penelitian memerlukan teori tentang java.

2.9 Android

Android merupakan sistem operasi yang memang khusus dirancang untuk smartphone dan tablet. Sistem android ini memiliki basis Linux yang mana dijadikan sebagai pondasi dasar dari sistem operasi android. Linux sendiri merupakan sistem operasi yang memang khusus dirancang untuk komputer. Android memang dirancang untuk dipasang pada perangkat-perangkat *mobile touchscreen* (*smartphone* dan *tablet*).

Sehingga sistem operasi yang berada di dalam smartphone saat ini memang menyesuaikan dari spesifikasi kelas *low-end* hingga *high-end*. Sehingga perkembangan sistem android memang cukup meningkat tajam. Android merupakan sistem operasi yang terbuka (*open source*) yang mana berarti jika pihak Google memperbolehkan dan membebaskan bagi pihak manapun untuk dapat mengembangkan sistem operasi tersebut. Bahkan anda sendiri pun juga dapat mengembangkan sistem android yang memang sesuai dengan keinginan anda.

Gambar 2. 2 Logo Android

Sistem Android memiliki gudang aplikasi dan game yaitu *Google Play Store*, yang mana disini anda bisa mendownload serta menggunakan aplikasi atau game yang terdapat di *Google Play Store* sepuasnya dengan menggunakan perangkat seluler dengan sistem android. Uniknya, Android menggunakan nama-nama makanan untuk membedakan versi sistem android yang diluncurkannya. Android menggunakan huruf depan dari nama makanan tersebut sebagai penanda peningkatan versi sistemnya. Mulai dari Cupcake Android 1.5 (C), Donuts Android 1.6 (D), Éclair Android 2.0-2.1 (E) atau Marshmallow Android 6.0 (M) hingga yang terbaru sekarang versi Pie Android 9.0(P).

Penelitian pembangunan aplikasi ini membahas tentang android dikarenakan banyak masyarakat menggunakan smartphone dan sistem operasi nya adalah android. Pembangunan aplikasi ini pun cocok diterapkan dalam android.

2.9.1 Sejarah Android

Telepon seluler menggunakan berbagai macam sistem operasi seperti *Symbian OS*®, *Microsoft's Windows Mobile*®, *Mobile Linux*®, *iPhone OS*® (berdasarkan *Mac OS X*), *Moblin*® (dari Intel), dan berbagai macam sistem operasi lainnya. API yang tersedia untuk mengembangkan aplikasi *mobile* terbatas dan oleh karena itulah Google mulai mengembangkan dirinya. *Platform* Android menjanjikan keterbukaan, kemudahan untuk menjangkau, *source code* yang terbuka, dan pengembangan *framework* yang *high end* .

Google membeli perusahaan Android Inc., yang merupakan sebuah perusahaan kecil berbasis pengembangan perangkat lunak untuk ponsel, pada tahun 2005 untuk memulai pengembangan pada *platform* Android. Tokoh utama pada Android Inc. meliputi Andy Rubin, Rich Miner, Nick Sears, dan Chris White.

Pada tanggal 5 November 2007, kelompok pemimpin industri bersama-sama membentuk *Open Handset Alliance* (OHA) yang diciptakan untuk mengembangkan standar terbuka bagi perangkat *mobile*. OHA terdiri dari 34 anggota besar dan beberapa anggota yang terkemuka diantaranya sebagai berikut: Sprint Nextel®, TMobile ®, Motorola®, Samsung®, Sony Ericsson®, Toshiba®, Vodafone®, Google, Intel® dan Texas Instruments.

Android SDK dirilis pertama kali pada 12 November 2007 dan para pengembang memiliki kesempatan untuk memberikan umpan balik dari pengembangan SDK tersebut. Pada bulan September 2008, T-Mobile memperkenalkan ketersediaan T-Mobile G1 yang merupakan *smart phone* pertama berbasiskan *platform* Android. Beberapa hari kemudian, Google merilis Android SDK 1.0. Google membuat *source code* dari *platform* Android menjadi tersedia di bawah lisensi *Apache's open source*.

Google merilis perangkat genggam (disebut Android Dev Phone 1) yang dapat menjalankan aplikasi Android tanpa terikat oleh berbagai jaringan *provider* telepon seluler pada akhir 2008. Tujuan dari perangkat ini adalah memungkinkan pengembang untuk melakukan percobaan dengan perangkat sebenarnya yang dapat menjalankan Android OS tanpa berbagai kontrak. Google juga merilis versi 1.1 dari sistem operasi Android pada waktu yang tidak lama. Versi 1.1 dari Android tidak mendukung adanya *soft keyboards* dan membutuhkan perangkat yang memiliki *keyboard* secara fisik. Android menyelesaikan masalah ini dengan merilis versi 1.5 pada bulan April 2009 dengan sejumlah tambahan fitur seperti kemampuan perekaman media,*widgets*, dan *live folders*.

Versi 1.6 dari Android OS dirilis pada bulan September 2009 dan hanya dalam waktu satu bulan versi Android 2.0 dirilis dan membanjiri seluruh perangkat Android. Versi ini memiliki kemampuan *advanced search*, *text to speech*, *gestures*, dan *multi touch*. Android 2.0 memperkenalkan kemampuan untuk menggunakan HTML karena didukung oleh HTML 5. Semakin banyak aplikasi berbasiskan Android setiap harinya yang terdapat pada *application store* secara *online* atau dikenal sebagai *Android Market*.

Sampai sekarang google telah meluncurkan beberapa versi android, yaitu :

1. Android 1.5 (*Cupcake*).
2. Android 1.6 (*Donut*).
3. Android 2.0/2.1 (*Eclair*).
4. Android 2.2 (*Froyo*).
5. Android 2.3 (*Gingerbread*).
6. Android 3.0 (*Honeycomb*).
7. Android 4.0 (*Ice Cream Sandwich*).
8. Android 4.1 (*Jelly Bean*).
9. Android 4.4 (*KitKat*).
10. Android 5.0 (*Lollipop*).
11. Android 6.0 (*Marshmallow*).

12. Android 7.0 (*Nougat*).
13. Android 8.0 (*Oreo*).
14. Android 9.0 (*Pie*).

2.9.2 Fitur Sistem Operasi Android

Sistem operasi Android memiliki fitur-fitur sebagai berikut:

1. Kerangka kerja aplikasi (*application framework*) Digunakan untuk menulis aplikasi di Android sehingga memungkinkan penggunaan kembali dan penggantian komponen. Kerangka kerja ini didukung oleh berbagai open source libraries seperti open ssl, sqlite,dan libc serta didukung oleh libraries utama Android. Kerangka kerja sistem operasi Android didasarkan pada UNIX file system permission yang menjamin bahwa aplikasi-aplikasi tersebut hanya memiliki kemampuan yang diberikan oleh pemilik ponsel pada waktu penginstalan.

2. *Dalvik Virtual Machine* (DVM)

   Dalvik Virtual Machine (DVM) adalah sebuah mesin virtual yang menggunakan memori yang sangat rendah dan secara khusus dirancang untuk Android untuk dijalankan pada *embedded system.* DVM bekerja dengan baik pada situasi dengan tenaga yang rendah dan mengoptimalkan perangkat *mobile*. DVM juga mengatur atribut dari *Central Processing Unit* (CPU) serta membuat sebuah format *file* yang spesial (.DEX) yang dibuat selama *build time post processing*. DVM mengambil *file* yang dihasilkan oleh *class* Java dan menggabungkannya ke dalam satu atau lebih *Dalvik Executable* (.dex).

   DVM dapat menggunakan kembali salinan informasi dari beberapa *class file* dan secara efektif mengurangi kebutuhan penyimpanan oleh setengah dari *Java Archive* (.jar) *file* tradisional. Konversi antara kelas Java dan format (.dex) dilakukan dengan memasukkan "dx tool*"*. DVM menggunakan *assembly-code*

yang berbeda dimana DVM menggunakan *register* sebagai unit utama dari penyimpanan data daripada menggunakan *stack*. Hasil akhir dari *executable-code* pada Android, merupakan hasil dari DVM yang didasarkan bukan pada Java *byte-code* melainkan pada file (.dex). Hal ini berarti bahwa Java *byte-code* tidak dieksekusi secara langsung melainkan dimulai dari Java *classfile* terlebih dahulu dan kemudian mengkonversikannya ke dalam *file* (.dex) yang berhubungan.

a. Browser yang terintegrasi
b. Grafik yang teroptimasi
c. SQLite
d. Dukungan media untuk suara, video dan format
e. GSM *Telephony* (bergantung dari perangkat keras yang digunakan)
f. *Bluetooth, Enhanced Data rate for GSM Evolution* (EDGE), *3rd Generation*(3G), dan WiFi (bergantung dari perangkat keras yang digunakan)
g. Kamera, *Global Positioning System* (GPS), kompas dan accelerometer(bergantung dari perangkat keras yang digunakan)
h. Lingkungan pengembangan yang lengkap, seperti emulator, peralatan untuk *debugging*, memori dan *performance*, *profiling*, serta *plug-in* untuk Eclipse IDE.

2.10 JSON

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generated*) oleh komputer. JSON terbuat dari dua struktur yaitu:

1. Kumpulan pasangan nama nilai. pada beberapa Bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel hash (*hash table*), daftar terkunci (*keyed list*), atau *associative array*.
2. Daftar nilai terurutkan (*list of value*). Pada kebanyakan Bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

JSON menggunakan bentuk sebagai berikut:

1. Objek

   Objek adalah sepasang nama/nilai yang tidak terurutkan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasangan nama/ nilai dipisahkan oleh, (koma).

2. Larik

   Larik adalah kumpulan nilai yang terurutkan. Larik dimulai dengan [(kurung kotak buka) dan diakhiri dengan] (kurung kotak tutup). Setiap nilai dipisahkan oleh “,” (koma).

3. Nilai

   Nilai (*value*) dapat berupa sebuah *string* dalam tanda kutip ganda, atau angka, atau true atau false atau null, atau sebuah objek atau sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat.

4. *String*

   String adalah kumpulan dari nol atau lebih karakter *Unicode*, yang dibungkus dengan tanda kutip ganda. Di dalam string dapat digunakan backslash escapes "\" untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada string. *String* sangat mirip dengan *string* C atau *Java*.

5. Angka, format oktal dan heksadesimal tidak digunakan.

2.11 MySQL

Menurut Abdul Kadir mdefinisikan bahwa : MySQL merupakan software yang tergolong database server dan bersifat open source. *Open source* menyatakan bahwa *software* ini di lengkapi oleh *source code* (kode yang dipakai untuk membuat MySQL), selai tentu saja bentuk *executable*-nya atau kode dapat di jalankan secara langsung di dalam sistem operasi, dan bisa diperoleh dengan cara mengunduh di internet secara gratis. Hal lainnya adalah MySQL jug bersifat *multiplatform*. MySQL dapat dijalankan pada berbagai jenis sistem operasi.

2.12 *Object Oriented Program* (OOP)

Object Oriented Program (OOP) merupakan paradigma baru dalam rekayasa *software* yang didasarkan pada obyek dan kelas. (Ronald J.N., 1996). Diakui para ahli bahwa *object-oriented* merupakan metodologi terbaik yang ada saat ini dalam rekayasa *software*. *Object-oriented* memandang *software* bagian per bagian dan menggambarkan satu bagian tersebut dalam satu obyek. Teknologi obyek menganalogikan sistem aplikasi seperti kehidupan nyata yang didominasi oleh obyek. Dengan demikian keunggulan teknologi obyek adalah bahwa model yang dibuat akan sangat mendekati dunia nyata yang masalahnya akan dipecahkan oleh sistem yang dibangun. Model obyek, atribut dan perlakuannya bisa langsung diambil dari obyek yang ada di dunia nyata.

Ada 4 (empat) prinsip dasar dari pemrograman berorientasi obyek yang menjadi dasar kemunculan UML, yaitu abstraksi, enkapsulasi, modularitas dan hirarki. Berikut dijelaskan satu persatu secara singkat.

1. Abstraksi memfokuskan perhatian pada karakteristik obyek yang paling penting dan paling dominan yang bisa digunakan untuk membedakan obyek tersebut dari obyek lainnya.
2. Enkapsulasi menyembunyikan banak hal yang terdapat dalam obyek yang tidak perlu diketahui oleh obyek lain.

3. Modularitas membagi sistem yang rumit menjadi bagian-bagian yang lebih kecil yang bias mempermudah *developer* memahami dan mengelola obyek tersebut.
4. Hirarki berhubungan dengan abstraksi dan modularitas, yaitu pembagian berdasarkan urutan dan pengelompokkan tertentu [8].

2.13 *Unified Modeling Language* (UML)

Unified Modelling Language (UML) adalah suatu alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara visual (Braun, et. al. 2001). Juga merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan objek (Whitten, et. al. 2004).

Sejarah UML sendiri terbagi dalam dua fase; sebelum dan sesudah munculnya UML. Dalam fase sebelum, UML sebenarnya sudah mulai diperkenalkan sejak tahun 1990an namun notasi yang dikembangkan oleh para ahli analisis dan desain berbeda-beda, sehingga dapat dikatakan belum memiliki standarisasi.

Fase kedua; dilandasi dengan pemikiran untuk mempersatukan metode tersebut dan dimotori oleh *Object Management Group* (OMG) maka pengembangan UML dimulai pada akhir tahun 1994 ketika Grady Booch dengan metode OOD (*Object-Oriented Design*), Jim Rumbaugh dengan metode OMT (*Object Modelling Technique*) mereka ini bekerja pada Rasional Software Corporation dan Ivar Jacobson dengan metode OOSE (*Object-Oriented Software Engineering*) yang bekerja pada perusahaan Objectory Rasional.

Sebagai pencetus metode-metode tersebut mereka bertiga berinisiatif untuk menciptakan bahasa pemodelan terpadu sehingga pada tahun 1996 mereka berhasil merilis UML versi 0.9 dan 0.91 melalui *Request for Proposal* (RFP) yang dikeluarkan oleh OMG (Braun, et.al. 2001). Kemudian pada Januari 1997 IBM, ObjecTime, Platinum Technology, Ptech, Taskon, Reich Technologies dan Softeam juga

menanggapi Request for Proposal (RFP) yang dikeluarkan oleh OMG tersebut dan menyatakan kesediaan untuk bergabung. Perusahaan-perusahaan ini menyumbangkan ide-ide mereka, dan bersama para mitra menghasilkan UML revisi 1.1. Fokus dari UML versi rilis 1.1 ini adalah untuk meningkatkan kejelasan UML Semantik versi rilis 1.0. Hingga saat ini UML versi terbaru adalah versi 2.0.

Saat ini sebagian besar para perancang sistem informasi dalam menggambarkan informasi dengan memanfaatkan UML diagram dengan tujuan utama untuk membantu tim proyek berkomunikasi, mengeksplorasi potensi desain, dan memvalidasi desain arsitektur perangkat lunak atau pembuat program.

Berdasarkan pembahasan diatas dapat disimpulkan bahwa UML bagian dari *Obejct Oriented Analysis* yang menitikberatkan analisis dari sisi pengguna atau orang-orang diluar sistem, bukan menitikberatkan pada aliran data di sistem.

2.13.1 Tujuan Penggunaan UML

Tujuan dari penggunaan diagram seperti diungkapan oleh Schmuller J. (2004), "*The purpose of the diagrams is to present multiple views of a system; this set of multiple views is called a model*" Berikut tujuan utama dalam desain UML adalah (Sugrue J. 2009) :

1. Menyediakan bagi pengguna (analisis dan desain sistem) suatu bahasa pemodelan visual yang ekspresif sehingga mereka dapat mengembangkan dan melakukan pertukaran model data yang bermakna.
2. Menyediakan mekanisme yang spesialisasi untuk memperluas konsep inti.
3. Karena merupakan bahasa pemodelan visual dalam proses pembangunannya maka UML bersifat independen terhadap bahasa pemrograman tertentu.
4. Memberikan dasar formal untuk pemahaman bahasa pemodelan.
5. Mendorong pertumbuhan pasar terhadap penggunaan alat desain sistem yang berorientasi objek (OO).

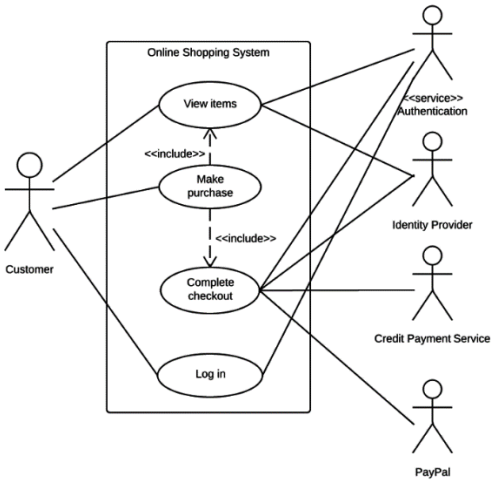
6. Mendukung konsep pembangunan tingkat yang lebih tinggi seperti kolaborasi, kerangka, pola dan komponen terhadap suatu sistem.
7. Memiliki integrasi praktik terbaik.

2.13.2 Jenis Diagram UML

Terdapat beberapa jenis diagram UML, yaitu :

1. *Use Case* Diagram

   Use case atau diagram *use case* merupakan pemodelan untuk kelakuan sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu [8].



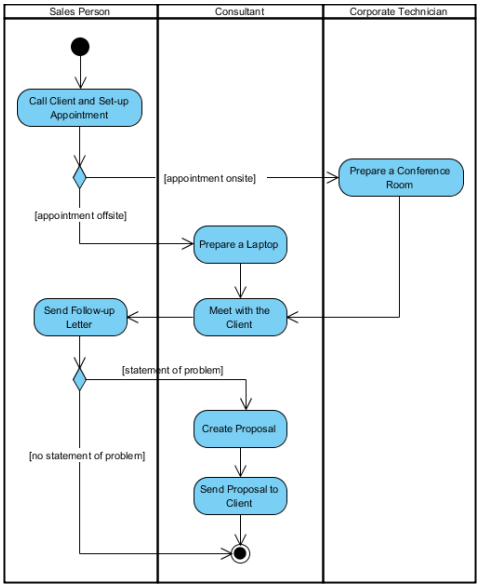
Gambar 2. 3 Contoh *Use Case* Diagram [8]

2. *Use Case Scenario*

Sebuah diagram yang menunjukkan *use case* dan aktor mungkin menjadi titik awal yang bagus, tetapi tidak memberikan detail yang cukup untuk desainer sistem untuk benar-benar memahami persis bagaimana sistem dapat terpenuhi. Cara terbaik untuk mengungkapkan informasi penting ini adalah dalam bentuk penggunaan *use case scenario* berbasis teks per u*se case*nya [8].

3. *Activity* Diagram

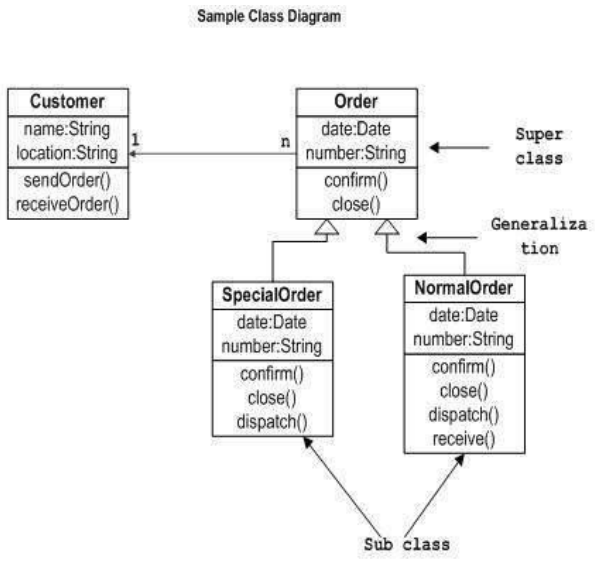
Diagram aktivitas atau *activity* diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem [8].



Gambar 2. 4 Contoh *Activity* Diagram [8]

4. *Class* Diagram

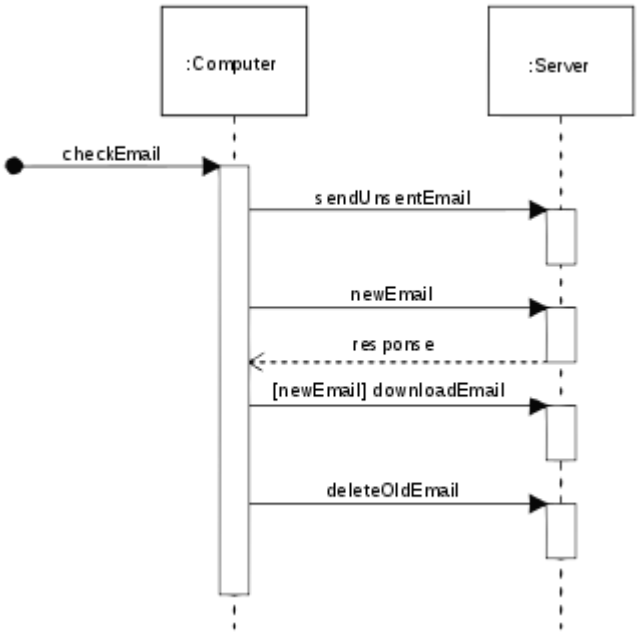
Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) satu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain.



Gambar 2. 5 Contoh *Class* Diagram [8]

5. *Sequence* Diagram

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan *sequence* diagram maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Membuat *sequence* diagram juga dibutuhkan untuk melihat skenario yang ada pada *use case* [8].



Gambar 2. 6 Contoh *Sequence* Diagram [8]