BAB III
LANDASAN TEORI

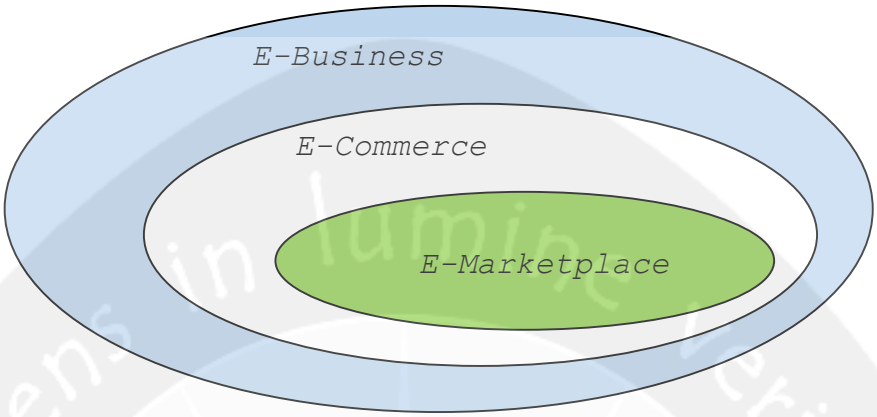
Pada bab ini akan dijelaskan mengenai teori dasar yang digunakan oleh penulis sebagai acuan dalam membangun aplikasi.

3.1 *E-Commerce*

E-commerce merupakan suatu kumpulan teknologi, aplikasi, dan proses bisnis yang menghubungkan perusahaan, konsumen, dan komunitas tertentu melalui transaksi elektronik dan perdagangan barang, pelayanan, dan informasi yang dilakukan secara elektronik (Witono & Hendrayana, 2011). Pendapat lain mendefinisikan *e-commerce* sebagai media membagikan informasi bisnis, mengelola relasi bisnis, dan mengalirkan transaksi bisnis dengan menggunakan jaringan telekomunikasi (Zwass, 1996). Suatu *e-commerce* juga memungkinkan organisasi atau perusahaan untuk mempromosikan produk dan layanan mereka secara *online* dan menyediakan berbagai layanan yang pelanggan sendiri dapat lakukan (Wiradinata, 2013).

E-commerce merupakan bagian dari *e-business* seperti terlihat pada gambar 3.1, di mana *e-business* menggunakan teknologi *internet* sebagai kekuatan utamanya untuk dapat melakukan beragam aktivitas bisnis secara elektronik yang efisien dan fleksibel (Pratama, 2005). Lee & Whang (2001) mendefinisikan *e-business* sebagai penggunaan *internet* dan komunikasi secara elektronik untuk mengeksekusi proses bisnis. Sedangkan menurut Laudon & Laudon (2006), mendefinisikan sebagai penggunaan

internet dan teknologi digital lainnya untuk komunikasi organisasi dan koordinasi pengelolaan perusahaan.



Gambar 3.1 Diagram Venn *E-Business* (Pratama, 2005)

Menurut Laudon & Laudon (2006), tiga model utama dari *e-commerce* adalah bisnis ke konsumen (B2C), bisnis ke bisnis (B2B), dan konsumen ke konsumen (C2C).

a. *Business to Business* (B2B) *e-commerce* melibatkan penjualan produk dan layanan antar perusahaan. Situs yang menerapkan B2B ini adalah ChemConnect, situs ini merupakan situs untuk membeli dan menjual gas alam cair, bahan bakar, bahan kimia, dan plastik.
b. *Business to Consumer* (B2C) *e-commerce* melibatkan penjualan produk dan layanan secara eceran kepada pembeli perorangan. Situs yang menerapkan B2C ini adalah Bhinneka.com, situs ini menjual peralatan elektronik, aksesoris komputer, alat tulis kantor, peralatan rumah tangga, piranti lunak, otomotif, dan lain sebagainya kepada konsumen perorangan.
c. *Consumer to Consumer* (C2C) *e-commerce* melibatkan konsumen yang menjual secara langsung ke konsumen. Sebagai contohnya eBay, tokopedia, OLX, situs tersebut memungkinkan orang-orang menjual barang mereka ke konsumen lain. Oleh karena itu, C2C ini

pula merupakan penerapan pada aplikasi *marketplace* yang dirancang.

3.2 E-Marketplace

E-marketplace didefinisikan sebagai retail atau pasar virtual secara *online* yang merupakan sub bagian dari *e-commerce* yang digunakan pada media elektronik antara pembeli dan penjual dalam melakukan transaksi jual beli, untuk keuntungan kedua belah pihak. Sebuah penyedia sistem *e-marketplace* membuat sistem infrastruktur perangkat keras dan perangkat lunak yang tersedia untuk memfasilitasi interaksi antara pembeli dan penjual (Hartmann, 2001). Fasilitas tersebut dapat berupa katalog produk sederhana dan dapat berupa pembayaran yang menjamin keamanannya (Sahney, 2008). Salah satu contoh dari *e-marketplace* adalah *website* tokopedia.com, *website* ini menyediakan dua jenis anggota di dalamnya, yaitu penjual dan pembeli. Transaksi pembayaran menggunakan transfer antar bank, dimana pihak tokopedia.com bertindak sebagai penengahnya, sehingga transaksi jual beli lebih aman.

3.3 Android

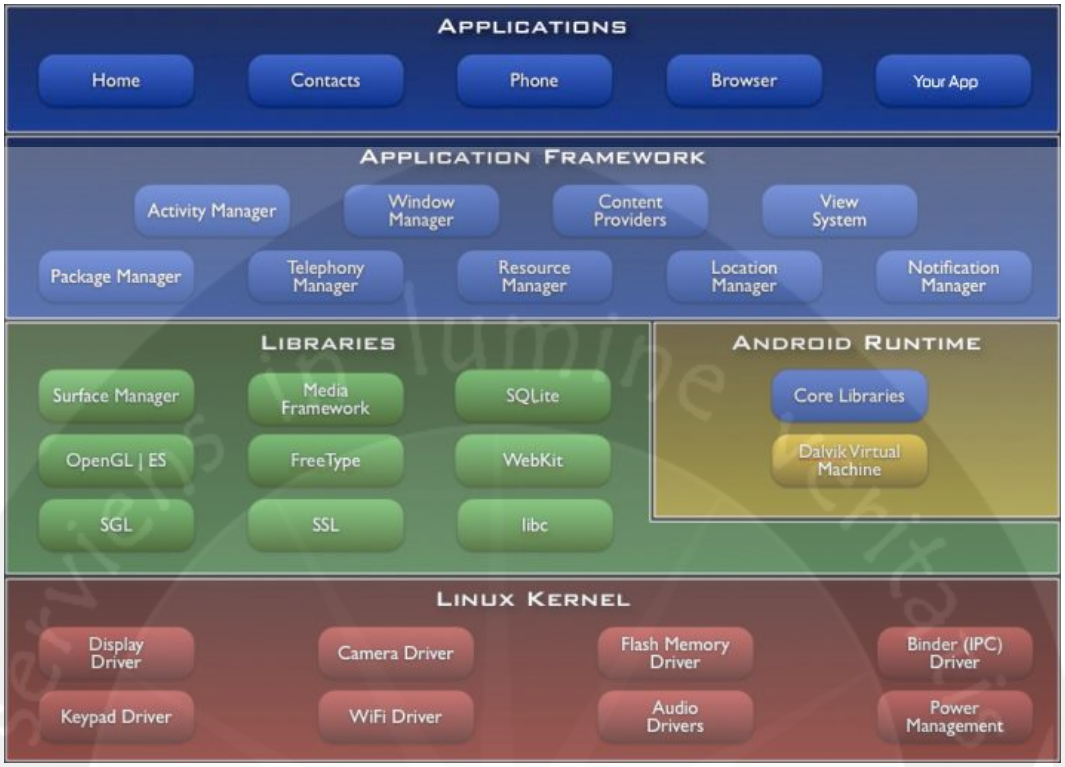
Android adalah sistem operasi untuk telepon seluler yang berbasis Linux. Android menyediakan *platform* terbuka (*open source*) bagi para pengembang untuk menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam piranti *mobile* (Nuryana & Putra, 2012). Awalnya, Google Inc. membeli Android Inc., pendatang baru yang membuat piranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah *Open Handset Alliance*, konsorsium dari 34 perusahaan piranti keras,

piranti lunak, dan telekomunikasi, termasuk Google, HTC, Intel, Motorola, Qualcom, Tmobile, Nvidia, dan lain-lain.

Arsitektur Android terdiri dari beberapa lapisan (Sariana, 2010), yaitu:

1. *Linux Kernel*
   Android bukanlah Linux, tetapi Android dibangun di atas Linux Kernel versi 2.6.
2. *Libraries*
   Android menyertakan satu set *libraries* C atau C++ yang digunakan dalam berbagai komponen sistem Android.
3. *Android Runtime*
   Android terdiri dari satu set perpustakaan inti (*core libraries*) yang menyediakan sebagian besar fungsi yang sama dengan yang terdapat dalam perpustakaan inti dari bahasa pemrograman Java.
4. *Application Framework*
   Arsitektur aplikasi dirancang agar komponen dapat digunakan kembali (*reuse*) dengan mudah.

5. *Application and Widget*
   Pada lapisan ini pengembang menempatkan aplikasi yang dikembangkan.

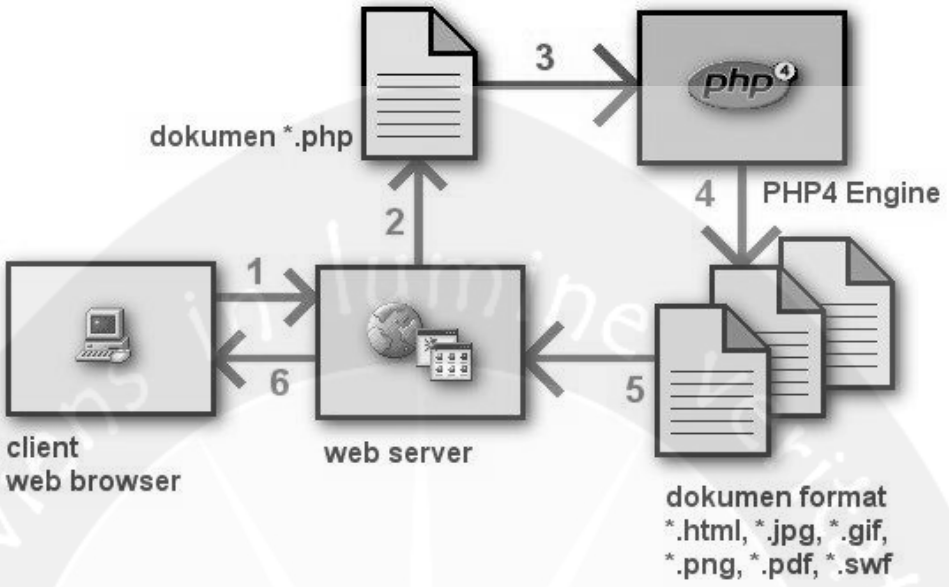


Gambar 3.2 Arsitektur Android (elinux.org)

3.4 *Server-Side Scripting*

Server-Side Scripting merupakan teknologi di mana *script* atau program dijalankan pada sisi *server*. Secara prinsip, *server* akan bekerja apabila ada permintaan dari *client*. Beberapa contoh dari server-side scripting, seperti: *Java Server Pages* (JSP), Perl, Phyton, ColdFusion, *Active Server Pages* (ASP), ASP.net, dan PHP.

Menurut Sariana (2010), cara kerja dari *server-side scripting* secara singkat dengan menggunakan PHP dapat dilihat pada gambar 3.3:



Gambar 3.3 Konsep Kerja *Server-Side Scripting* (Sariana, 2010)

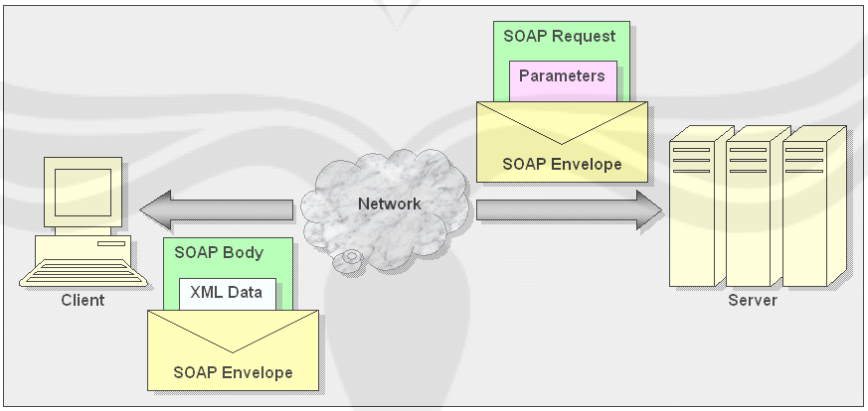
Keterangan:

A. *Client* mengirim *request* ke *web server* melalui *browser*.
B. *Web server* menerima *request* dalam bentuk dokumen PHP.
C. Berkas PHP dikirimkan ke PHP *engine* untuk diproses.
D. PHP *engine* menerjemahkan berkas PHP menjadi kode HTML.
E. Setelah melalui proses, berkas kembali dikirim ke *web server*.
F. *Web server* mengirimkan berkas ke *client* untuk ditampilkan pada *browser* sehingga bisa dilihat oleh pengguna.

3.5 Web Service

Web service adalah kumpulan fungsi atau *method* yang terdapat pada sebuah *web server* yang dapat dipanggil oleh klien dari jarak jauh (Marthasari, et al., 2010). *Web service* dapat menjalankan operasi-operasi termasuk akses data, update data pada *database*, mengirim respon dari *request* yang dikirim oleh *client*, sehingga operasi-operasi yang dimiliki oleh *web service* nantinya dapat digunakan oleh aplikasi lainnya yang mengirimkan *request* (Anindito, 2012).

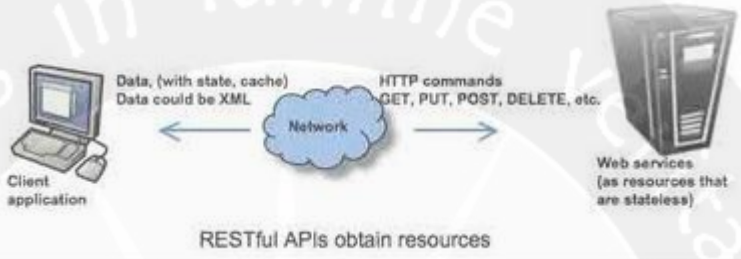
Contoh teknologi dari *web service*, yaitu SOAP dan REST. W3C (2007) mendefinisikan SOAP (*Simple Object Access Protocol*) sebagai salah satu standarisasi dasar *web service* untuk bertukar informasi yang ditulis dengan format XML. Sedangkan REST atau *Representational State Transfer* merupakan salah satu standarisasi dasar *web service* yang dapat ditulis dalam format *JavaScript Object Notation*(JSON) atau pun XML (Mitchell, 2013).



Gambar 3.4 Alur proses akses *web service* dengan SOAP (devx.com)

Alur proses akses *web service* SOAP seperti pada gambar 3.4 bahwa ketika *client* mengakses *web service* SOAP maka *client* akan mengirimkan SOAP *request* beserta parameternya yang kemudian akan dibungkus dengan

standarisasi SOAP. Setelah itu *request* akan dikirimkan ke *web server* menggunakan media jaraingan, lalu data akan diproses dan hasil dari proses *request* akan dikirimkan kembali ke *client* melalui media jaringan, selanjutnya *client* akan menerima respon *web service* SOAP yang berisi SOAP *body* beserta data dalam bentuk XML.



Gambar 3.5 Alur proses *web service* dengan REST (Mitchell, 2013)

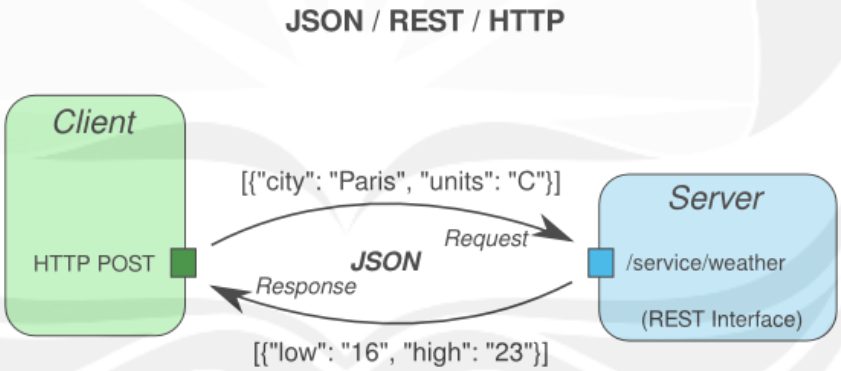
Pada gambar 3.5 menunjukkan alur proses *request web service* menggunakan REST. Ketika *client* mengirimkan *request* beserta parameternya ke *web server* maka *request web service* akan diproses dan hasilnya dikirimkan ke *client* dalam format yang telah ditentukan pada *web service* yang dapat bebentuk format XML atau pun JSON.

Tabel 3.1 Perbandingan antara *Web Service* SOAP dan REST (Wagh & Thool, 2012)

SOAP	REST
Teknologi lama	Teknologi baru
Interaksi *Client-Server* sangat bergantungan	Interaksi *Client-Server* kurang bergantungan
Hanya *support* format XML	*Support* tipe format seperti JSON, XML
Membutuhkan *tools* pengembangan	Tidak membutuhkan *tools* pengembangan

3.6 *Java Script Object Notation* (JSON)

Java Script Object Notation (JSON) merupakan format penulisan pertukaran data layaknya *Extended Markup Language* (XML). JSON mampu melakukan pemidahan data antara dua *interface* dengan sangat cepat dan *powerfull* (misalnya antara PHP dengan JavaScript). Format JSON tidak tergantung dengan bahasa pemrograman apapun, struktur JSON sederhana sehingga mudah diimplementasikan. Karena JSON lebih sedikit membutuhkan *space* dan tidak perlu dituliskan dengan lengkap layaknya XML. Sehingga secara logika, proses pengolahannya (*parsing*) lebih cepat (Fatmanto, 2013). Contoh penggunaan JSON pada aplikasi TanioMall dapat dilihat pada gambar 3.6:



Gambar 3.6 Penggunaan JSON Rest (linkeddataorchestration.com)

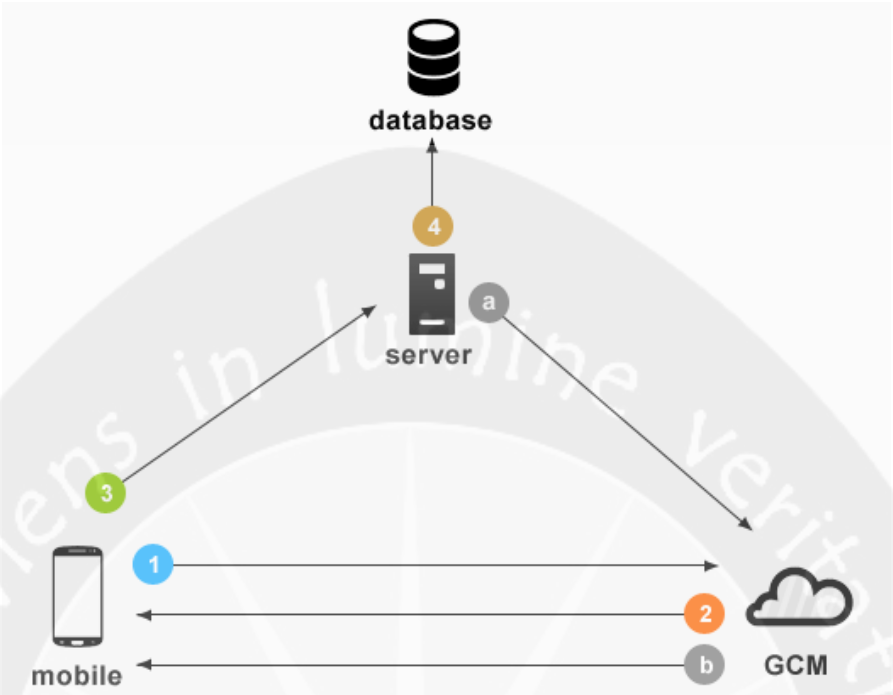
Pada gambar 3.6 merupakan contoh *client* mengirimkan *request* data ke *server* menggunakan method POST. Data yang dikirimkan berupa *city*: Paris dan *units*: C. Lalu pada *server* akan menjalankan fungsi "/service/weather" biasanya fungsi ini berupa *query* PHP. Lalu fungsi akan mengembalikan data low: 16 dan high: 23 ke *client*, kemudian *client* akan membaca *response* data melalui layar ponsel.

3.7 *Google Cloud Messaging* (GCM)

Google Cloud Messaging (GCM) untuk Android merupakan sebuah layanan (*service*) yang mengijinkan aplikasi untuk mengirimkan data dari *server* kepada pengguna Android, dan juga menerima pesan dari perangkat pada koneksi yang sama. Layanan GCM menangani pesan dan pengiriman ke aplikasi Android yang dituju yang berjalan pada perangkat target pengiriman notifikasi.

Dalam prosesnya, GCM membutuhkan komponen-komponen sebagai berikut:

1. Perangkat *mobile*, perangkat yang menjalankan aplikasi Android yang menggunakan GCM. Perangkat ini haru memiliki versi Android minimal 2.3 (*Gingerbread*), dan harus memiliki minimal satu *login account* Google, misalnya akun *Google Play Service*.
2. *Third-party Application Server*, sebuah server yang dirancang oleh pengembang sebagai bagian dari proses GCM dalam aplikasi mereka. *Server* ini mengirim data ke aplikasi Android melalui server GCM (*Google Cloud Messaging*). *Server* ini juga harus mengimplementasikan teknik *exponential back-off* dimana teknik ini yang menangani proses dari GCM jika terjadi kesalahan.
3. GCM *server*, server Google yang terlibat dalam pengambilan pesan dari *Third-party Application Server* dan mengirimnya ke aplikasi Android.



Gambar 3.7 Alur Kerja GCM (developer.android.com)

Pada gambar 3.7 menunjukkan alur kerja GCM. Berikut alur pendaftaran GCM:

1. Pertama perangkat Android mengirimkan *sender id* dan *application id* menuju *server* GCM untuk registrasi.
2. Setelah registrasi pada *server* GCM maka *server* tersebut akan mengirimkan *registration id* pada perangkat android.
3. Setelah menerima *registration id*, perangkat akan megirimkan *registration id* menuju *server* aplikasi.
4. *Server* aplikasi akan menyimpan *registration id* pada *database*.

Alur pengiriman notifikasi menggunakan GCM, sebagai berikut:

a. Ketika membutuhkan notifikasi, *server* aplikasi mengirimkan sebuah pesan ke *server* GCM beserta *registration id* perangkat yang tersimpan pada *database server* aplikasi.

b. *Server* GCM menerima pesan yang selanjutnya diteruskan kepada perangkat Android pengguna yang memiliki *registration id* yang sesuai.

Pada bab landasan teori ini telah dibahas mengenai teori yang digunakan oleh penulis sebagai pedoman dalam membangun aplikasi. Pada bab selanjutnya, yaitu bab analisis dan perancangan perangkat lunak, akan dijelaskan mengenai analisa latar belakang pembangunan sistem, analisa sistem yang akan dibangun, dan perancangan sistem.