BAB II

TINJAUAN PUSTAKA

II.1. Penelitian Terdahulu

Demi kesempurnaan penelitian, maka penulis perlu melakukan perbandingan untuk mengetahui gambaran dari penelitian terdahulu. Menurut M Zia Ulhaq, (2010 : Hal 1-2), dalam jurnal nya yang berjudul "Rancang Bangun Aplikasi Pengisian Pulsa Elektrik Berbasis Android J2ME Pada *Flow Celluler*", Pada penelitian ini, akan dilakukan pembangunan aplikasi pengisian pulsa elektrik berbasis j2me dengan biaya seminim mungkin, guna menunjang kinerja aplikasi tiap hari dalam implementasinya.

Menurut Hengki Mulyono, Rodiah (2013 : Hal 17-35), dalam jurnal nya yang berjudul"Implementasi Algoritma *One Time Pad* Penyimpanan Data Berbasis Web", Penelitian ini akan mengimplementasikan algoritma *One Time Pad* (OTP) untuk melakukan penyandian terhadap data dan informasi yang disimpan,

Menurut Rizki Putra Mustofa, (2013 : Hal 14), dalam jurnal nya yang berjudul "Aplikasi Mobile Android *One Time Pad* (OTP) Untuk meningkatkan Keamanan Otentikasi" Password dari sistem ini memiliki waktu kadaluarsa dengan cara terus berganti setiap kali login sehingga meningkatkan keamanan suatu akun dan meminimalisir terjadinya serangan yang dilakukan *hacker* atau *cracker*,.

Menurut Zuhar Musliyana, *and et all*, (2016 : Hal 21), dalam jurnal nya yang berjudul “Peningkatan Sistem Keamanan Otentikasi *Single Sign On* (SSO) Menggunakan Algoritma AES dan *OneTime Password* Studi Kasus: SSO Universitas Ubudiyah Indonesia”, Sistem ini berjalan pada *Protocol Hypertext Transfer Protocol* (HTTP).Penelitian ini mengusulkan penggunaan algoritma *Advanced Encryption Standard* (AES) dengan pembangkit kunci dinamis dan metode *One Time Password* (OTP) berbasis sinkronisasi waktu dengan kombinasi *salt* untuk meningkatkan keamanan pada otentikasi,

Berdasarkan penelitian terdahulu, maka dilakukan penelitian selanjutnya yaitu “Perancangan Aplikasi Pengisian Pulsa Elektronik Berbasis Android Dengan Menggunakan Metode Simple OTP (*One Time Pad*)” guna untuk membangun aplikasi yang sama hanya saja lebih dikembangkan sistem opersinya. Tujuan dari penelitian ini untuk sama dengan penelitian terdahulunya yaituuntukmemberi kemudahan dalam melakukan transaksi serta aplikasi ini mampu meminimalkan kesalahan pada saat transaksi dan mampu menjaga keamanan sistem.

II.2. Landasan Teori

II.2.1. Sejarah Kriptografi

Kriptografi sebagian besar merupakan sejarah kriptografi klasik, yaitu metode enkripsi yang menggunakan kertas dan pensil atau mungkin dengan bantuan alat mekanik sederhana. Secara umum algoritma kriptografi klasik dikelompokkan menjadi dua kategori, yaitu algoritma transposisi (*transposition*

cipher) dan algoritma substitusi (*substitution cipher*). *Cipher* transposisi mengubah susunan huruf-huruf di dalam pesan, sedangkan *cipher* substitusi mengganti setiap huruf atau kelompok huruf dengan sebuah huruf atau kelompok huruf lain, (Fresly Nandar Pabokory *and ett all* : 2015 : 22).

II.2.1.1. Kriptografi

Kriptografi adalah ilmu dan seni untuk menjaga kerahasian pesan dengan cara menyandikannya kedalam bentuk yang tidak dapat dimengerti lagi maknanya. Dalam ilmu kriftografi, terdapat dua buah proses yaitu melakukan enkripsi dan deskripsi. Pesan yang akan dienkripsi disebut sebagai *plaintext* (teks biasa). Disebut demikian karena informasi ini dengan mudah dapat dibaca dan dipahami oleh siapa saja. Algoritma yang dipakai untuk mengenkripsi dan mendekripsi sebuah *plaintext* melibatkan penggunaan suatu bentuk kunci. Pesan *plaintext* yang telah dienkripsi (atau dikodekan) dikenal sebagai *ciphertext* (teks sandi), (Fresly Nandar Pabokory *and ett all* : 2015 : 22).

Di dalam kriptografi kita akan sering menemukan berbagai istilah atau *terminology*. Berberapa istilah yang harus diketahui yaitu :

1. Pesan *(Massage)*

Pesan adalah data dan informasi yang dapat dibaca dan dimengerti maknanya. Nama lain untuk pesan adalah (*plaintext*) atau teks jelas (*cleartext*).

2. Pengirim dan Penerima

Komunikasi data melibatkan pertukaran pesan antara dua entitas. Pengirim (*sender*) adalah entitas yang mengirim pesan kepada entitas lainnya. Penerima (*receiver*) adalah entitas yang menerima pesan.

3. Enkripsi dan dekripsi

Proses menyandikan *plainteks* menjadi *cipherteks* disebut enkripsi (*encryption*) atau *enciphering* (standard nama menurut ISO 7498-2). Sedangkan proses mengembalikan *cipherteks* menjadi plainteks semula disebut dekripsi (*decryption*) atau *deciphering* (standard nama menurut ISO 7498-2).

4. *Cipher* dan kunci

Algoritma kriptogarfi disebut juga *cipher*, yaitu aturan untuk enkripsi dan dekripsi, atau fungsi matematika yang digunakan untuk enkripsi dan dekripsi. Beberapa *cipher* memerlukan algoritma yang berbeda untuk enkripsi dan dekripsi. Konsep matematis yang mendasari algoritma kriptografi adalah relasi antara dua buah himpunan yang berisi elemen-elemen *plainteks* dan himpunan yang berisi *cipherteks.* Enkripsi dan dekripsi merupakan fungsi yang memetakan elemen-elemen antara dua himpunan tersebut. Misalkan *P* menyatakan *plainteks* dan *C* menyatakan *cipherteks*, maka :

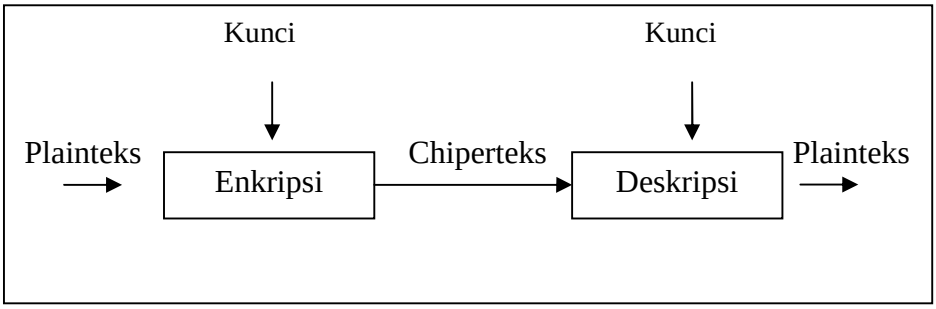
$E(P) = C \rightarrow$ fungsi enkripsi *E* memetakan *P* ke *C*.

$D(C) = P \rightarrow$ fungsi dekripsi *D* memetakan *C* ke *P*.

Karena proses enkripsi kemudian dekripsi mengembalikan pesan ke pesan asal, maka persamaan $D(E(P)) = P$ harus benar.

Kriptografi mengatasi masalah keamanan data dengan menggunakan kunci, yang dalam hal ini algoritma tidak dirahasiakan lagi, tetapi kunci harus tetap dijaga kerahasiaannya. Kunci (*key*) adalah parameter yang digunakan untuk transformasi enkripsi dan dekripsi. Kunci biasanya berupa *string* atau deretan bilangan.

Dengan menggunakan kunci *K*, maka fungsi enkripsi dan dekripsi dapat ditulis sebagai skema diperlihatkan pada Gambar II.1.



Gambar II.1. Skema Enkripsi Dan Dekripsi Dengan Menggunakan Kunci

II.2.1.2. Tujuan kriptografi

Dari paparan awal dapat dirangkumkan bahwa kriptografi bertujuan untuk member layanan keamanan, yang dinamakan aspek-aspek keamanan:

1. Kerahasiaan (*confidentiality*)

   Adalah layanan yang ditujukan untuk menjaga agar pesan tidak dapat dibaca oleh pihak-pihak yang tidak berhak.

2. Integritas data (*data integrity*)

   Adalah layanan yang menjamin bahwa pesan masih asli atau belum pernah dimanipulasi selama pengiriman.

3. Otentikasi (*authentication*)

   Adalah layanan yang berhubungan dengan identifikasi, baik mengidentifikasi kebenaran pihak-pihak yang berkomunikasi (*user autehentication*).

4. *Non-repudiation*

   Adalah layanan untuk menjaga entitas yang berkomunikasi melakukan penyangkalan.

II.2.2. Pulsa Elektronik

Pengertian pulsa dapat diartikan sebagai alat perhitungan atau sistem perhitungan dalam menentukan tarif pelanggan. Fungsi pulsa adalah sebagai satuan biaya untuk melakukan komuniksai atau telepon, mengirim (sms), *chatting*, *massenger* dan bahkan untuk bermain *game online,* (Icuk Bagus Setya :2015 : Hal 22).

Ada 2 jenis pengisian pulsa yaitu fisik (*voucher*) dan elektronik (*elektrik*). Untuk pengisian pulsa fisik, yaitu dengan cara menggosok salah satu bagian tertentu pada *voucher*, kemudian menulis kode *voucher* dan melakukan panggilan secara langsung. Sedangkan pengisian pulsa elektronik dilakukan dengan cara pelanggan datang secara langsung ke kios pulsa kemudian penjual pulsa akan melakukan transaksi dengan mengetik format SMS dan mengirim ke no SMS center pulsa. Dan mayoritas pelanggan lebih condong terhadap pengisian pulsa secara elektronik daripada pengisian pulsa secara fisik.Alasan utama adalah kemudahan dan harga yang lebihmurah.

Dalam proses pengisian pulsa elektronik (*elektrik*) ada 2 metode, yaitu satu *chip* satu operator dan satu *chip* semua operator. Untuk metode satu *chip* satu operator digunakan khusus untuk pengisian dari produk provider tersebut. Misalnya *chip* M-Kios digunakan khusus untuk pengisian pulsa Simpati dan As. Sedangkan metode satu *chip* semua operator digunakan khusus untuk proses pengisian pulsa semua operator yang berbeda-beda misalnya XL, Mentari, IM3, dan beberapa operator lainnya yang bisa bertransaksi dalam satu *chip* tersebut. (M Zia Ulhaq ; 2010 : Hal 1-2).

II.2.3. *Smartphone*

Smartphone adalah telepon genggam yang mempunyai kemampuan menyerupai komputer, dengan kata lain *smartphone* ini mempunyai fungsi hampir sama layaknya komputer, namun dalam bentuk yang lebih kompak layaknya sebuah telepon seluler. Selama ini belum ada pengertian mutlak, apa itu *smartphone*. Sebagian orang berpendapat bahwa *smartphone* adalah telepon genggam yang bekerja dengan menggunakan seluruh piranti lunak sistem operasi yang menyediakan hubungan standar dan mendasar bagi pengembang aplikasi. (Eko Zunantono 2013 : Hal 2).

II.2.4. *One Time Pad* (OTP)

Menurut Ariyus, Doni, (2008 : Hal 82) dalam buku nya yang berjudul “Pengantar Ilmu KRIPTOGRAFI” bahwa One *Time Pad* (OTP)berisi deretan kunci yang dibangkitkan secara acak. Pertama ditemukan oleh Mayor J

maugborne dan G vernam pada tahun 1917, setiap kunci hanya digunakan untuk sekali pakai. Pemilihan kunci harus secara acak agar tidak bisa diproduksi ulang dan membuat lawan tidak mudah menerka dan jumlah karakter kunci sama dengan jumlah karakter yang dimiliki pesan, contoh; jika diketahui plaintext "KEAMANAN" maka, kuncinya bisa dipakai "ZJKEOLFH".

One Time Pad (OTP) adalah *password* yang berlaku hanya untuk satu sesi *login* atau transaksi. OTP tidak rentan terhadap serangan *replay*. Ini berarti jika penyusup potensial berhasil merekam OTP yang sudah digunakan untuk masuk ke layanan atau untuk melakukan transaksi, penyusup tidak akan dapat menyalah gunakannya karena tidak berlaku lagi.

Jumlah kunci sama panjangnya dengan jumlah *plainteks*. Jika anda ingin agar *cipherteks* sulit untuk di pecahkan maka pemakaian kunci seharusnya :

1. Jangan gunakan kunci yang berulang.
2. Pilihkan kunci yang *random*.

Metode penyandian OTP merupakan salah satu variasi dari metode penyandian substitusi dengan cara memberikan syarat-syarat khusus terhadap kunci yang digunakan yaitu terbuat dari huruf yang acak atau bilangan yang acak (kunci acak atau *pad*), dan pengacakannya tidak menggunakan rumus tertentu. Kunci tersebut benar-benar acak, digunakan hanya sekali, serta terjaga kerahasiannya dengan baik, maka metode penyandian OTP ini sangat kuat dan tidak mudah untuk dipecahkan.

Keterangan :

E (Enkripsi) : Sebuah proses menjadikan pesan yang dapat dibaca *(plainteks)* menjadi pesan acak yang tidak dapat dibaca *(chiperteks)*.

D (Deskripsi) : Proses kebalikan dari enkripsi dimana proses ini akan mengubah *chiperteks* menjadi *plainteks.*

P (*Plainteks*) : Pesan yang dapat dibaca.

C (*Cipherteks*) : Pesan acak yang tidak dapat dibaca.

K (Key) : Kunci untuk melakukan teknik kriptografi.

X (Simbol) : Berisi tentang sederetan abjad atau nilai dari abjad tersebut.

Rumus melakukan OTP dengan menggunakan sederetan huruf / abjad yaitu untuk memudahkan dalam operasional, huruf-huruf diterjemahkan dahulu kedalam angka 1 sampai 26 dengan A = 1, B = 2; dst sampai Z = 26. Dan dalam perhitungan aljabarnya berupa bilangan modulus 26.

Untuk memudahkan pemahaman, bisa diperhatikan contoh berikut :

Rumus melakukan OTP yaitu :

Enkripsi : $E(x) = (P(x) + K(x)) \ \text{Mod } 26$

Dekripsi : $D(x) = (C(x) - K(x)) \ \text{Mod } 26$

Pesan yang akan disandi :

DUNIA

Kunci acak :

FGHJV

Algoritma : Teks Sandi / *Chippertext* = Teks Asli / *Plainteks* + Kunci

Proses :

Penjumlahan ini menggunakan bilangan modulus 26.

Teks Asli / Plainteks : D=4 U=21 N=14 I=9 A=1

Kunci : F=6 G=7 H=8 J=10 V=22

Teks Sandi / *Chiperteks* : (4+6)=10=J (21+27-26)=2=B (14+8)=22=V

(9+10)=19=S (1+22)=23=W

Teks Hasil Penyandian : JBVSW

Setelah melihat contoh rumus metode OTP diatas maka penulis menemukan cara yang lebih mudah untuk dimengerti dan lebih *simple* dalam penerapan metode *simple* OTP *(One Time Pad)* Pada Perancangan Aplikasi Pengisian Pulsa Elektronik Berbasis Android yaitu dengan menggunakan sederatan angka .

Rumus melakukan OTP yaitu :

Enkripsi : E(x) = (P(x) + K(x))

Dekripsi : D(x) = (C(x) – K(x))

Contoh Enkripsi Pesan :

Pesan : **4 3 2 1**

Kunci : **1 1 2 2**

Algoritma : Teks sandi / *Chippertext* = *Plainteks* + Kunci

Maka perhatikan langkahnya seperti di bawah ini :

Plainteks : 4321

Kunci : 1122 (+)

Hasil Penyandian : 5443

Jadi *Chiperteks* yang di hasilkan yaitu : **5443**

Deskripsi pesan, perhatikan langkah di bawah ini :

Algoritma : Teks sandi / *Plainteks* = *Chiperteks* - Kunci

Chiperteks : 5443

Kunci : 1122 (-)

Hasil Penyandian : 4321

Jadi *Plainteks* yang di hasilkan yaitu : **4321**

Metode penyandian OTP ini kekuatannya bertumpu pada keacakan kuncinya, sehingga kunci yang digunakan untuk proses penyandian tersebut harus dilindungi dengan baik.

II.2.5. *UML*

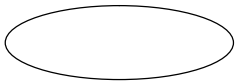
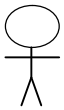
UML adalah bahasa spesifikasi standar yang dipergunkan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan, umum adalah industri perangkat lunak dari pengembangan sistem. (Windu Gata : 2013)

II.2.5.1.Diagram-*Diagram UML*

1. *Diagram Use Case (Use Case Diagram)*

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.1. Simbol *Use Case Diagram*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case.*
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	*Extend,* merupakan perluasan dari *use case* lain jika

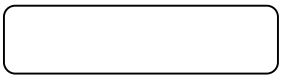
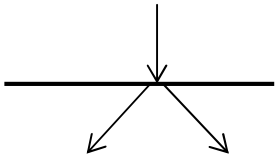
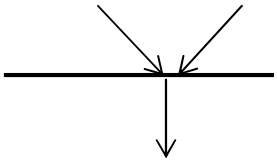
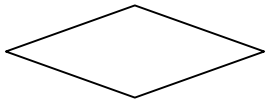
	kondisi atau syarat terpenuhi.

Sumber : (Windu Gata : 2013)

2. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram,* yaitu :

Tabel II.2. Simbol Diagram Aktivitas

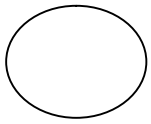
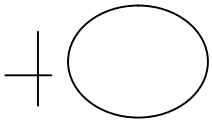
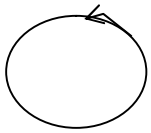
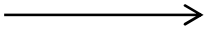
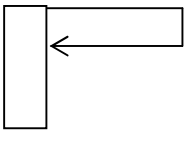
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau *rake*, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true, false.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

Sumber : (Windu Gata : 2014)

3. *Sequence Diagram*

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel II.3. Simbol *Sequence* Diagram

Gambar	Keterangan
	*Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan *formentry* dan *form* cetak.
	*Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*, *activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.

	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

Sumber : (Windu Gata : 2013)

4. *Class Diagram* (Diagram Kelas)

Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.

Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan.

Class diagram secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.4. Simbol *Class* Diagram

Multiplicity	**Penjelasan**
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

Sumber : (Windu Gata : 2013)

II.2.6. Android

Menurut I Gusti Rai Agung Sugiarta, I Wayan Kardana (2013 : Hal : 296), Android merupakan sistem operasi yang dijalankan di perangkat *mobile* yang bersifat *open source*. Dibeli oleh *Google* yang dimiliki oleh *Open Handset Alliance*. Tujuan utamanya adalah “Berinovasi dalam perangkat *mobile*, banyak pengguna, murah dan pengalaman penggunaan *mobile* yang lebih baik.”Android merupakan sistem operasi terdiri dari beberapa lapisan (*layers*). Setiap lapisan memiliki ciri khusus dan fungsi tersendiri.

II.2.6.1. Kelebihan Android

Sudah banyak *platform* untuk perangkat *celular* saat ini, termasuk didalamnya *Symbian, iPhone, Windows Mobile, Blacberry, Java Mobile Edition, Linux Mobile (Limo),* dan banyak lagi. Namun ada beberapa hal yang menjadi kelebihan Android. Walaupun beberapa fitur-fitur yang ada telah muncul sebelumnya pada platform lain, Android adalah yang pertama menggabungkan hal seperti berikut :

1. Keterbukaan, Bebas pengembangan tanpa dikenakan biaya terhadap sistem karena berbasiskan Linux dan *open source*. Pembuat perangkat menyukai hal ini karena dapat membangun *platform* yang sesuai yang diinginkan tanpa harus membayar *royality*. Sementara pengembangan *software* menyukai karena Android dapat digunakan diperangkat manapun dan tanpa terikat oleh vendor manapun.

2. Arsitektur komponen dasar android terinspirasi dari teknologi internet *Mashup*. Bagian dalam sebuah aplikasi dapat digunakan oleh aplikasi lainnya, bahkan dapat diganti dengan komponen lain yang sesuai dengan aplikasi yang dikembangkan.
3. Banyak dukungan *service*, kemudahan dalam menggunakan berbagai macam layanan pada aplikasi seperti penggunaan layanan pencarian lokasi, database SQL, *browser* dan penggunaan peta. Semua itu sudah tertanam pada Android sehingga memudahkan dalam pengembangan aplikasi.
4. Siklus hidup aplikasi diatur secara otomatis, setiap program terjaga antara satu sama lain oleh berbagai lapisan keamanan, sehingga kerja sistem menjadi lebih stabil. Pengguna tak perlu khawatir dalam menggunakan aplikasi pada perangkat yang memorinya terbatas.

II.2.7. SQLite

Menurut I Gusti Rai Agung Sugiarta, I Wayan Kardana (2013 : Hal : 296), SQLite adalah basis data e*mbedded* (tertanam) disuatu perangkat keras. Berjalan dengan proses mandiri yang berdampingan dengan jalannya aplikasi yang dilayaniya. Kode basis data saling terkait, atau tertanam sebagai kesatuan dengan aplikasi. Bagi sebagian besar pengamat tidak akan pernah melihat basis data ini sebagai *Relational Database Management* (RDBMS). Aplikasi hanya melakukan pekerjaan dan mengelola data.

II.2.8. Basic4Android

Basic4android adalah *development tool* sederhana yang *powerful* untuk membangun aplikasi Android. Bahasa Basic4android mirip dengan Visual Basic dengan tambahan dukungan untuk objek.Aplikasi Android (APK) yang dicompile oleh Basic4Android adalah aplikasi Android *native*/asli dan tidak ada extra *runtime* seperti di Visual Basic yang ketergantungan file msvbvm60.dll, yang pasti aplikasi yang dicompile oleh Basic4Android adalah *NO DEPENDENCIES* (tidak ketergantungan file lain). IDE Basic4Android hanya fokus pada *development* Android, (Press Penny, Seagrave Wyken : 2013 : Hal 195).