BAB II

TINJAUAN PUSTAKA

2.1 Penjualan

Penjualan adalah bagian dari promosi dan promosi adalah salah satu bagian dari keseluruhan sistem pemasaran (Abdullah, 2016:3).

Jenis-Jenis Penjualan Penjualan memiliki beberapa jenis penjualan, diantaranya adalah sebagai berikut :

1. PenjualanTunai

   Pengertian Penjualan Tunai menurut mengatakan bahwa: Penjualan tunai dilaksanakan oleh perusahaan dengan cara mewajibkan pembeli melakukan pembayaran harga barang terlebih dahulu sebelum barang diserahkan oleh perusahaan kepada pembeli (Mulyadi, 2013:455).

2. PenjualanKredit

   Pengertian Penjualan Tunai menurut mengatakan bahwa: Penjualan tunai dilaksanakan oleh perusahaan dengan cara mewajibkan pembeli melakukan pembayaran harga barang terlebih dahulu sebelum barang diserahkan oleh perusahaan kepada pembeli (Mulyadi, 2013:455).

3. Penjualan Online

   Penjualan atau Pemasaran Online adalah proses menjual produk atau jasa menggunakan media internet atau jaringan, artinya kegiatan pemasaran dilakukan secara elektronik lewat internet atau jaringan cyber (Priyan:2016).

2.2 Aplikasi

Aplikasi berasal dari kata *Application* yang artinya penerapan, penggunaan. Secara istilah aplikasi adalah program siap pakai yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain dan dapat digunakna oleh sasaran yang dituju. Menurut *Whitten* proses dimana keperluan pengguna dirubah kedalam bentuk paket perangkat lunak atau kedalam spesifikasi pada komputer yang berdasarkan pada sistem informasi (Rizka, 2013)

2.3 Sistem Operasi Android

2.3.1 Pengertian Android

Android merupakan *subset* perangkat lunak untuk perangkat *mobile* yang meliputi sistem operasi, *middleware* dan aplikasi untuk yang di *release* oleh *google*. Android adalah sistem operasi untuk telepon seluler yang berbasis *linux*. Android menyediakan *platform* terbuka bagi para pengembang buat menciptakan aplikasi mereka sendiri untuk digunakan oleh bermacam peranti bergerak. Awalnya *Google Inc*, membeli Android *Inc*, pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan Android, dibentuklah *Open Headset Alliance,* Konsorsium dari 34 perusahaan peranti keras, peranti lunak, dan telekomunikasi, termasuk *Google, HTC, Intel, Motorola, Qualcomm, T-Mobile,* dan *Nvidia* (Widi, 2017).

2.3.2 Android Studio

Android Studio adalah Lingkungan Pengembangan Terpadu – Intergrated Development Environment (IDE) untuk pengembangan aplikasi Android, berdasarkan IntelliJ IDEA. Selain merupakan editor kode IntelliJ dan alat pengembang yang berdaya guna, Android Studio menawarkan fitur lebih banyak untuk meningkatkan produktivitas saat membuat aplikasi Android, misalnya :

Sistem pembuatan berbasis Gradle yang fleksibel, emulator yang cepat dan kaya fitur. Lingkungan yang menyatu untuk pengembangan bagi semua perangkat Android. Instant Run untuk mendorong perubahan ke aplikasi yang berjalan tanpa membuat APK baru. Template kode dan integrasi GitHub untuk membuat fitur aplikasi yang sama dan mengimpor kode contoh. Alat penguji dan kerangka kerja ekstensif. Alat Lint untuk meningkatkan kinerja, kegunaa, kompatibilitas versi dan masalah – masalah lain. Dukungan C++ dan NDK dukungan bawaan untuk Google Cloud Platform, mempermudah pengintegrasian Google Cloud Messaging dan App Engine(Pusitasari, 2017).

2.3.3 *Android* SDK (*Software Development Kit*)

Android SDK (*Sofware Development Kit*) adalah *tools* dan API yang diperlukan untuk mengembangkan aplikasi pada *platform* Android dengan menggunakan bahasa pemograman java (Nazaruddin,2014).

Beberapa fitur-fitur Android yang paling penting adalah (Nazaruddin, 2014):

1. *Framework* aplikasi yang mendukung penggantian komponen atau *reusable*.
2. Mesin *Virtual Dalvik* dioptimalkan untuk perangkat *mobile*.
3. *Integrated browser* berdasarkan *engine opensourceWebKit*.
4. *Grafis* yang dioptimalkan dan didukung oleh *libraries grafis* 2D, *grafis* 3D berdasarkan spesifikasi opengl ES 1,0 (Operasional Akselerasi *Hardware*).
5. *SQLite* untuk penyimpanan data.
6. Media *support* yang mendukung audio, video, dan gambar (MPEG4, H.264, MP3, AAC, AMR, JPG, PNG, GIF), GSM *Telephony* (tergantung *hardware*).
7. *Bluetooth, EDGE, 3G,* dan *Wi-Fi* (tergantung *hardware*).
8. Kamera, GPS, kompas, dan *accelerometer* (tergantung *hardware*).
9. Lingkungan *Development* yang lengkap dan kaya termasuk perangkat *emulator*, *tools* untuk *debugging*, profil dan kinerja memori.

2.3.4 Jenis-jenis Versi Android

Berikut ini merupakan perkembangan Android dari versi ke versi.Dapat dilihat pada Tabel 2.1 di bawah ini.

No.	Versi	Nama	Tanggal Rilis	Level
1.	1.0	Tanpa Nama	23 September 2009	1
2.	1.1	Tanpa Nama	9 Februari 2009	2
3.	1.5	*Cupcake*	30 April 2009	3
4.	1.6	*Donut*	15 September 2009	4
5.	2.0	*'Eclair*	26 Oktober 2009	5
6.	2.1	*'Eclair*	12 Januari 2010	7
7.	2.2	*Froyo*	20 Mei 2010	8
8.	2.3	*Gingerbread*	6 Desember 2010	10
9.	3.0	*Honeycomb*	22 Februari 2011	11
10.	3.1	*Honeycomb*	10 Mei 2011	12

11.	3.2	*Honeycomb*	15 Juli 2011	13
12.	4.0	*Ice Cream Sandwich*	19 Oktober 2011	14 15
13.	4.1	*Jelly Bean*	9 Juli 2012	16
14.	*4.2*	*Jelly Bean*	13 November 2012	17
15.	4.4	*KitKat*	November 2013	18
16.	5.5	*Lollipop*	Mei 2014	19
17.	6.0	*Marshmallow*	5 Oktober 2015	20
18.	7.0	*Nougat*	Agustus 2016	21
19.	8.0	*Oreo*	Maret 2017	22

Tabel 2.1 Versi Android

2.4 Database

Database (basis data) adalah kumpulan file – file yang mempunyai kaitan antara satu file dengan file lainnya sehingga membentuk data untuk menginformasikan satu perusahaan dan instansi. Bila terdapat file yang tidak dapat dihubungkan dengan file yang lainnya, berarti file tersebut bukanlah kelompok dari satu *Database,* melainkan membentuk satu *database* sendiri. *Database* juga merupakan landasan bagi pembuatan dan pengembangan program aplikasi. Oleh karena itu, *database* harus dibuat sedemikian rupa sehingga pembuatan program lebih mudah dan cepat (Tiara, 2013).

Definisi Dasar :

1. *Database* (Basis Data): Sekumpulan data yang saling berhubungan untuk mencapai suatu tujuan.
2. *Data*: kumpulan yang berisi fakta-fakta, yang disimpan ditempat tertentu.
3. *Tabel* : Tempat untuk menyimpan data, tabel terdiri dari field dan record.
4. *Field*(Kolom): salah satu bagian tabel untuk menyimpan item data.
5. *Record*(Baris): salah satu bagian tabel diman terdapat satu bagian informasi yang disimpan dalam tabel.

2.5 Aplikasi Pendukung

2.5.1 Firebase

Firebase adalah layanan DbaaS (Database as a Service) dengan konsep realtime. Firebase merupakan penyedia layanan cloud dengan backend sebagai servis yang berbasis di San Fransisco, California. Firebase terdiri dari fitur pelengkap yang bisa dipadupadankan sesuai dengan kebutuhan Anda. Firebase memberikan anda perlengkapan dan infrastuktur untuk membangun suatu aplikasi yang lebih baik dan meningkatkan kesuksesan bisnis anda.Produk utama dari Firebase yaitu suatu database yang menyediakan API untuk memungkinkan pengembang menyimpan dan mensinkronisasi data lewat multiple client.Firebase menyediakan library untukberbagai client platform. Untuk browser menggunakanJavascript dan untuk mobile menggunakan Objective-C atau Android API (Nadira:2016).

2.5.2 SQLite

SQLite merupakan sebuah sistem manajemen basisdata relasional yang bersifat ACID-compliant dan memiliki ukuran pustaka kode yang relatif kecil, ditulis dalam bahasa C. SQLite merupakan proyek yang bersifat public domain yang dikerjakan oleh D. Richard Hipp. – WikipediaJadi gampangnya, SQLite ini adalah mesin database SQL yang tertanam pada sistem yang kita gunakan. Tidak seperti pada paradigma client-server umumnya, Inti SQLite bukanlah sebuah sistem yang mandiri yang berkomunikasi dengan sebuah program, melainkan sebagai bagian integral dari sebuah program secara keseluruhan.Sehingga protokol komunikasi utama yang digunakan adalah melalui pemanggilan API secara langsung melalui bahasa pemrograman. Mekanisme seperti ini tentunya membawa keuntungan karena dapat mereduksi overhead, latency times, dan secara keseluruhan lebih sederhana. Seluruh elemen basisdata (definisi data, tabel, indeks, dan data) disimpan sebagai sebuah file. SQLite dapat digunakan di Windows Phone, Android, iPhone, PHP, Firefox, Chrome dan lain – lain yang dapat dilihat di: http://www.sqlite.org/famous.html. Pada browser, biasanya SQLite ini digunakan untuk menyimpan kongfigurasi seperti history, bookmark,

dan cache, sedangkan pada mobile, penggunaan SQLite ini sangat banyak seperti kontak, database tabel dsb (Angon:2016).

2.5.3 JSON

JSON (JavaScript Object Notation) merupakan format yang ringan untuk memasukan data ke dalam sebuah variabel. Sangat mudah dimengerti dan diimplementasikan oleh manusia, dan mudah juga untuk komputer dalam melakukan parsingnya.JSON merupakan bagian dari bahasa pemrograman JavaScript (Standard ECMA-262 3rd Edition – December 1999). JSON merupakan format teks yang sepenuhnya independen tetapi menggunakan konvensi yang familiar dengan bahasa pemrograman dari keluarga-C, termasuk C, C++, C#, Java, JavaScript, Perl, Python, dan sebagainya.

Kelebihan inilah yang membuat JSON menjadi sebuah bahasa data-interchange yang ideal. JSON dibangun dalam dua struktur, Beberapa pasangan dari nama/nilai. Dalam beberapa bahasa perograman biasa disebut dengan istilah object, record, struct, tabel hash, key list atau associative array. Nilai-nilai yang terusun secara ordered list. Biasa disebut dengan array, vector, list atau daftar dalam bahasa pemrograman.

JSON adalah struktur data yang universal, dalam artian bisa digunakan dalam berbagai bahasa pemrograman. Hampir semua bahasa pemrograman mendukung penuh JSON dalam berbagai format. Hal ini memungkinkan format data yang dapat dipertukarkan menggunakan bahasa pemrograman juga menggunakan dasar dari struktur JSON(Yusro:2013).

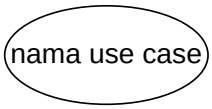
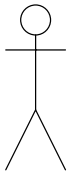
2.5.4 *Unified Modeling Language* (UML)

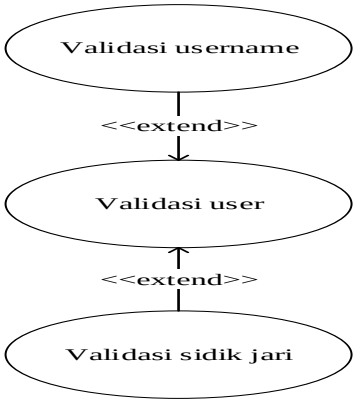
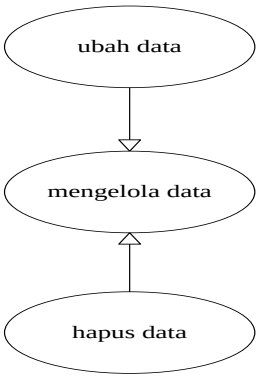
Unified Modelling Language (UML) adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Untuk merancang sebuah model, UML memiliki beberapa diagram antara lain use case diagram, class diagram, statechart

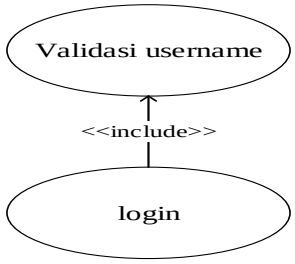
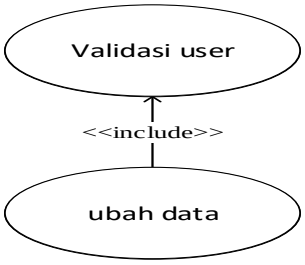
diagram, activity diagram, sequence diagram, collaboration diagram, component diagram, deployment diagram (Widodo, 2015).

2.5.5 Use Case Diagram

Use case diagram merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan tentang interaksi antara satu aktor atau lebih dengan sistem informasi yang akan dibuat. *Use case* biasanya digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi yang akan dibangun dan siapa saja yang dapat menggunakan fungsi-fungsi tersebut (Rosa dan Shalahuddin, 2013, h.155). Simbol-simbol yang ada pada *use case diagram* (Rosa dan Shalahuddin, 2013, h.156) dapat dilihat pada Tabel 2.2 dibawah ini.

No	Simbol	Deskripsi
1	*Use case* 	Fungsioalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*.
2	Aktor/*actor* 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor.
3	Asosiasi/*association* ———	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor.
4	Ekstensi/*extend* ——→	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu; mirip

No	Simbol	Deskripsi
		dengan prinsip *inheritance* pada pemrograman berorientasi objek; biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case y*ang ditambahkan, misal :  Arah panah mengarah pada *use case* yang ditambahkan; biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya.
5	Generalisasi/ *generalization* 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi satu adalah fungsi yang lebih umum dari lainnya, misalnya: 

No	Simbol	Deskripsi
6	Menggunakan/ *include*/i <<include>> «uses»	Merupakan relasi *use case* tambahan ke sebuah *use case*. *Use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankannya *use case* ini. Ada dua sudut pandang yang cukup besar mengenai *include* pada *use case*: *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan, misal pada kasus berikut:  *Include* berarti *use case* tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan, misal pada kasus berikut:  Kedua interpretasi di atas dapat dianut salah satu atau keduanya tergantung pada pertimbangan dan interpretasi yang dibutuhkan.

Tabel 2.2 Simbol *Use Case*

2.5.5.1 Manfaat *Use Case*

Adapun manfaat dari *Use Case* yaitu:

1. Digunakan untuk berkomunikasi dengan end user dan domain expert.
2. Memastikan pemahaman yang tepat tentang requirement / kebutuhan sistem.
3. Digunakan untuk mengidentifikasi siapa yang berinteraksi dengan sistem.

2.5.5.2 Karakteristik *Use Case*

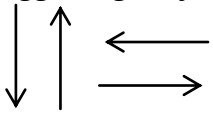
Use cases adalah interaksi atau dialog antara sistem dan actor, termasuk pertukaran pesan dan tindakan yang dilakukan oleh sistem.

1. Use cases diprakarsai oleh actor dan mungkin melibatkan peran actor lain. Use cases harus menyediakan nilai minimal kepada satu actor.
2. Use cases bisa memiliki perluasan yang mendefinisikan tindakan khusus dalam interaksi atau use case lain mungkin disisipkan.
3. Use case class memiliki objek use case yang disebut skenario. Skenario menyatakan urutan pesan dan tindakan tunggal.

2.5.6 Activity Diagram

Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (alirankerja) atau aktivitas dari sebuah system atau proses bisnis. Yang perlu diperhatikan bahwa diagram aktivitas menggambarkan aktivitas system bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan sistem. (Rosa A.S – Shalahuddin, 2013, h.161).Pada Tabel 2.3 dibawah ini merupakan simbol-simbol yang ada pada diagram Activity.

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.

Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan / *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Simbol	**Deskripsi**
Penggabungan / *join* 	Digunakan untuk menghubungkan satu simbol denga symbol lainnya
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.

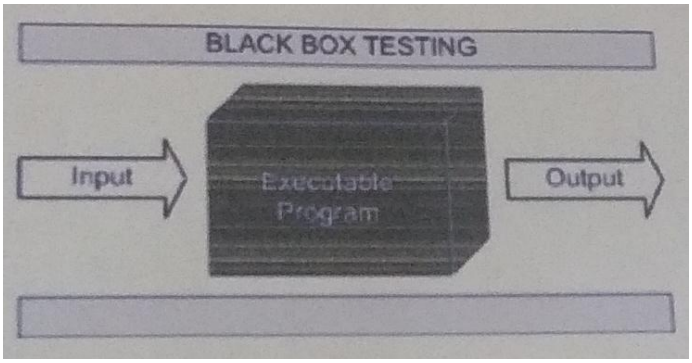
Tabel 2.3 Simbol *Activity Diagram*

2.3 Teori Pengujian Sistem

Menurut Fattah (2007:171) pengujian unit digunakan untuk menguji setiap modul dan menjamin setiap modul menjalankan fungsinya dengan baik.

Menurut Simarmata (2009) mengatakan pengujian adalah sebuah proses terhadap aplikasi atau program untuk menemukan segala kesalahan dan segala kemungkinan yang akan menimbulkan kesalahan sesuai dengan spesifikasi perangkat lunak yang telah ditentukan sebelum aplikasi tersebut diserahkan kepada pengguna. Salah satu jenis metode untuk melakukan pengujian pada perangkat lunak yaitu *black box testing.*

Black box testing terfokus pada apakah unit program memenuhi kebutuhan (*requirement*) yang disebutkan daam spesifikasi. Pada *black box testing,* Cara pengujian dilakukan dengan menjalankan atau mengeksekusi unit atau modul, kemudian diamati apakah hasil dari unit ini sesuai dengan proses yang diinginkan.



Gambar 2.1 *BlackBox Testing*

Teknik yang digunakan dalam *black box testing* antara lain :

a) Digunakan untuk menguji fungsi-fungsi khusus dari perangkat lunak.
b) Kebenaran perangkat lunak yang diuji hanya dilihat berdasarkan keluaran (*output*) yang dihasilkan.
c) Kemampuan program dalam memenuhi kebutuhan pemakai dapat diukur sekaligus dapat diketahui kesalahan-kesalahannya.