BAB II

LANDASAN TEORI

2.1. Tinjauan Pustaka

A. Konsep Dasar Sistem Informasi

Menurut Hutahaen (2014:13) mengemukakan bahwa "Sistem informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengelolaan transaksi harian, mendukung operasi, bersifat manajerial, dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang dibutuhkan."

Menurut Laudon dan Laudon (2007:15) "Sistem informasi (*information system*) secara teknis dapat didefinisikan sebagai sekumpulan komponen yang saling berhubungan, mengumpulkan (atau mendapatkan), memproses, menyimpan, dan mendistribusikan informasi untuk menunjang pengambilan keputusan dan pengawasan dalam suatu organisasi."

Menurut Kusrini dan Koniyo (2007:9) "Definisi umum sistem informasi adalah sebuah sistem yang terdiri atas rangkaian subsistem informasi terhadap pengolahan data untuk menghasilkan informasi yang berguna dalam pengambilan keputusan."

Dari beberapa pendapat diatas maka dapat disimpulkan bahwa sistem informasi adalah sistem yang terdiri dari rangkaian informasi sehingga dapat

dikembangkan untuk tujuan berbeda tergantung pada kebutuhan bisnis dan berguna untuk pengambilan keputusan.

Supaya sistem itu dikatakan sistem yang baik memiliki karakteristik menurut Hutahaen (2014:3), yaitu:

1. Komponen Sistem (*Components*)
   Suatu sistem terdiri dari sejumlah komponen-komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan.
2. Batasan Sistem (*Boundary*)
   Batasan sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lain atau dengan lingkungan luarnya.
3. Lingkungan Luar Sistem (*Environment*)
   Lingkungan luar sistem (*environment*) adalah diluar batas dari sistem yang mempengaruhi operasi sistem.
4. Penghubung Sistem (*Interface*)
   Penghubung sistem merupakan media penghubung antara satu subsistem dengan subsistem lainnya.
5. Masukan Sistem (*Input*)
   Masukan adalah energi yang dimasukkan kedalam sistem, yang dapat berupa perawatan (*maintenance input*), dan masukan sinyal *(signal input*).
6. Keluaran Sistem (*Output*)
   Keluaran sistem adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.
7. Pengolah Sistem (*Process*)
   Suatu sistem menjadi bagian pengolah yang akan merubah masukan menjadi keluaran.
8. Sasaran Sistem (*Objective*)
   Suatu sistem pasti mempunyai tujuan (*goal*) atau sasaran (*objective*).

B. E-Commerce

Menurut Wong (2010:33) “Pengertian dari *electronic commerce* adalah pembelian, penjualan dan pemasaran barang serta jasa melalui sistem elektronik. Seperti televisi, radio dan jaringan komputer atau *internet*.” *E-commerce* meliputi

transfer dana secara elektronik, pertukaran dan pengumpulan data. Semua diatur dalam sistem manajemen inventori otomatis.

Menurut Hidayat (2008:10) "*e-commerce* atau electronic commerce adalah bagian dari e-lifestyle yang memungkinkan traksaksi jual-beli dilakukan secara online dari sudut tempat manapun."

Klasifikasi *e-commerce* menurut Syarif (2016:69) adalah sebagai berikut :

1. *Business to Business* (B2B)
   E-commerce tipe ini meliputi transaksi antar organisasi yang dilakukan di *Electronic market.*
2. *Business to Customer* (B2C)
   Merupakan transaksi eceran dengan pembeli perorangan.
3. *Customer to Customer* (C2C)
   Konsumen menjual secara langsung ke konsumen lain. Atau mengiklankan jasa pribadi di *Internet*.
4. *Customer to Business* (C2B)
   Perseorangan yang menjual produk atau layanan ke organisasi, perseorangan yang mencari penjual, berinteraksi dan menyepakati suatu transaksi.
5. *Nonbusiness e-commerce*
   Lembaga non bisnis seperti akademis, organisasi, organisasi keagamaan, organisasi sosial dan lembaga pemerintahan yang menggunakan berbagai tipe *e-commerce* untuk mengurangi biaya guna meningkatkan operasi dan layanan publik.
6. *Intrabusiness (organizational) e-commerce*
   Termasuk kategori ini adalah semua aktivitas intern organisasi, biasanya dijalankan di *internet* yang melibatkan pertukaran barang, jasa/informasi.

Sejumlah orang memandang istilah *commerce* (perdagangan) sebagai transaksi yang dilakukan antar perusahaan yang berpartner. Karena itu istilah *electronic commerce* berkesan sempit bagi sejumlah orang. Demikianlah, banyak yang lebih suka menggunakan istilah *e-business*, yang mengacu pada definisi *e-*

commerce secara lebih luas, tidak sekedar menjual dan membeli, namun juga berarti melayani pelanggan dan berkolaborasi dengan partner bisnis, serta pelaksanaan traksaksi elektronik dalam suatu organisasi.

C. Unified Modeling Language (UML)

Menurut Mulyani (2016:42) "*Unified Modeling Language* (UML) adalah sebuah teknik pengembangan sistem yang menggunakan bahasa grafis sebagai alat untuk pendokumentasian dan melakukan spesifikasi pada sistem." UML adalah metodologi untuk mengembangkan sistem OOP dan sekelompok *tool* untuk mendukung pengembangan sistem tersebut.

Menurut Muslihudin dan Oktafianto (2016:58) "UML memiliki arti bahasa pemodelan standar yang memiliki sintaks dan semantik serta mempunyai aturan-aturan yang harus diikuti." Dengan menggunakan UML, desain *software* dapat diwujudkan dalam bentuk simbol dan diagram. Desain dalam bentuk simbol dan diagram, kemudian dapat diterjemahkan menjadi kode program.

Menurut Nugroho (2010:6) "UML (*Unified Modeling Language*) adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma berorientasi objek." Pemodelan (*modeling*) sesungguhnya digunakan utnuk penyederhanaan permasalah-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami.

1. Use Case Diagram

Menurut Nugroho (2009:7) "*Use case diagram* merupakan deskripsi lengkap tentang interaksi yang terjadi antara para aktor dengan sistem/perangkat lunak yang sedang kita kembangkan."

Menurut Triandini dan Suardika (2012:17) "*Use case* adalah sebuah kegiatan yang dilakukan oleh sistem, biasanya dalam menanaggapi permintaan dari pengguna sistem."

Menurut Rosa A.S dan M. Shalahuddin (2013:155) "*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.

Sebuah *use case* merepresentasikan sebuah sistem interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya *login* ke sistem, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *test case* untuk semua *feature* yang ada pada sistem.

2. Activity Diagram

Menurut Muslihudin dan Oktafianto (2016:63) "Diagram aktifitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem."

Menurut Nugroho (2009:13) "Keunggulan dari *activity diagram* adalah bahwa diagram tersebut lebih mudah dipahami dibandingkan skenario. Selain itu,

dengan menggunakan *activity diagram*, kita juga bisa melihat di bagian manakah sistem dari suatu skenario akan berjalan."

Menurut Rosa A.S dan M. Shalahuddin (2013:161) "Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak."

Activity diagram mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku *parallel* sedangkan *flowchart* tidak bisa. *Activity diagram* merupakan *state diagram* khusus, dimana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan *behavior internal* sebuah sistem dan interaksi antar subsistem secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum. Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. Sama seperti *state*, standard UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behavior* pada kondisi tertentu, digambarkan dengan simbol belah ketupat. Untuk mengilustrasikan proses *parallel* (*fork and join*) digunakan titik singkronisasi yang dapat berupa titik, garis horizontal atau vertikal. *Activity diagram* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.

3. *Component Diagram*

Menurut Muslihudin dan Oktafianto (2016:63) "Diagram komponen memperlihatkan organisasi serta ketergantungan sistem/perangkat lunak pada komponen-komponen yang telah ada sebelumnya."

Menurut Rosa A.S dan M. Shalahuddin (2013:148) "Diagram komponen atau *component diagram* dibuat untuk menunjukkan organisasi dan ketergantungan diantara kumpulan komponen dalam sebuah sistem."

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) diantara komponen. Komponen piranti lunak adalah model berisi kode, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time, link time,* maupun *run time*. Umumnya komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen-komponen yang lebih kecil. Komponen dapat juga berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.

4. *Deployment Diagram*

Menurut Muslihudin dan Oktafianto (2016:63) "*Diagram deployment* berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponen-komponen."

Menurut Rosa A.S dan M. Shalahuddin (2013:154) "*Diagram deployment* atau *deployment* diagram menunjukkan konfigurasi komponen dalam proses eksekusi aplikasi."

Deployment diagram juga dapat digunakan untuk memodelkan sistem tambahan (*embedded system*) yang menggambarkan rancangan *device*, *node* dan *hardware*, sistem *client/server*, sistem terdistribusi murni dan rekayasa ulang aplikasi.

D. Entity Relationship Diagram (ERD)

Menurut Fatansyah (2012:81) "Model *Entity-Relationship* yang berisi komponen-komponen Himpunan Entitas dan Himpunan Relasi yang masing-masing dilengkapi dengan atribut-atribut yang merepresentasikan seluruh fakta dari dunia nyata yang kita tinjau, dapat digambarkan dengan lebih sistematis dengan menggunakan *Diagram Entity-Relationship* (Diagram E-R)."

Menurut Rosa A.S dan M. Shalahuddin (2013:50) "ERD adalah bentuk paling awal dalam melakukan perancangan basis data relasional. Jika menggunakan OODBMS maka perancangan ERD tidak perlu dilakukan."

Menurut Fatansyah (2012:81) Notasi simbolik di dalam Diagram E-R yang dapat digunakan sebagai berikut :

1. Persegi panjang, menyatakan Himpunan Entitas.
2. Lingkaran/Elip, menyatakan Atribut (Atribut yang berfungsi sebagai *key* digarisbawahi).
3. Belah Ketupat, menyatakan Himpunan Relasi.
4. Garis, sebagai penghubung antara Himpunan Relasi dengan Himpunan Entitas dan Himpunan Entitas dengan Atributnya.
5. Kardinalitas Relasi dapat dinyatakan dengan banyaknya garis cabang atau dengan pemakaian angka (1 dan 1 untuk relasi satu-ke-satu, dan N untuk relasi satu-ke-banyak atau N dan N relasi banyak-ke-banyak).

Menurut Fatansyah (2012:86) tahapan dalam pembuatan diagram E-R, yaitu :

1. Mengidentifikasi dan menetapkan seluruh himpunan entitas yang akan terlibat
2. Menentukan atribut-atribut key dari masing-masing himpunan entitas
3. Mengidentifikasi dan menetapkan seluruh himpunan relasi di antara himpunan entitas-himpunan entitas yang ada beserta *foreign-key*-nya
4. Menentukan derajat/kardinalitas relasi untuk setiap himpunan relasi
5. Melengkapi himpunan entitas dan himpunan relasi dengan atribut-atribut deskriptif (non *key*)

E. Logical Record Structure (LRS)

Menurut Frieyadie (2007:13) "LRS merupakan hasil dari pemodelan *Entity Relationship* (ER) beserta atributnya sehingga bisa terlihat hubungan-hubungan antar entitas."

Menurut Kroenke (2007:76) "*Logical Record Structure* (LRS) adalah representasi dari struktur record-record pada tabel-tabel yang terbentuk dari hasil relasi antar himpunan entitas."

Dalam pembuatan LRS terdapat 3 hal yang dapat mempengaruhi menurut Frieyadie (2007:13) yaitu :

1. Jika tingkat hubungan (*cardinality*) satu pada satu *(one-to-one*), maka digabungkan dengan entitas yang lebih kuat (*strong entity*), atau digabungkan dengan entitas yang memiliki atribut yang lebih sedikit).
2. Jika tingkat hubungan (*cardinality*) satu pada banyak (*one-to-many*), maka hubungan relasi atau digabungkan dengan entitas yang tingkat hubungannya banyak.
3. Jika tingkat hubungan (*cardinality*) banyak pada banyak (*many-to-many*), maka hubungan relasi tidak akan digabungkan dengan entitas manapun, melainkan menjadi sebuah LRS.

2.2. Penelitan Terkait

Dalam penulisan skripsi ini, penulis memasukan dua jurnal yang berhubungan dengan pembahasan yang diambil, yaitu :

A. Menurut Nugroho (2016:717) menjelaskan bahwa : "Dengan *e-commerce* telah banyak merubah dalam proses jual-beli. Jika dalam suatu jual-beli penjual dan pembeli bertemu, namun jika dengan *e-commerce* mereka tidak perlu bertemu, mereka berinteraksi dengan melalui internet maupun dengan komunikasi melalui telepon atau *chatting*."
B. Menurut Hastanti, dkk (2015:1) menjelaskan bahwa : "Mempromosikan produk dengan menggunakan website akan lebih banyak menguntungkan juga mempermudah proses pengembangan dan dapat menghemat biaya. Konsumen lebih mudah memilih produk tanpa harus data langsung."

Berdasarkan jurnal diatas penulis akan membuat Sistem Informasi e-commerce untuk menunjang kinerja dari Toko Fotocopy Nicky dalam memberikan pelayanan terhadap seluruh masyarakat.