BAB II

LANDASAN TEORI

2.1 Konsep Dasar Sistem

2.1.1 Pengertian Sistem

Menurut Kadir (2008), sistem adalah sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan. Sebagai gambaran, jika dalam sebuah sistem terdapat elemen yang tidak memberikan manfaat dalam mencapai tujuan yang sama, maka elemen tersebut dapat dipastikan bukanlah bagian dari sistem.

Menurut Mulyanto (2009), sistem adalah sekelompok komponen yang saling berhubungan, bekerjasama untuk mencapai tujuan bersama dengan menerima *input* serta menghasilkan *ouput* dalam proses transformasi yang teratur. Kumpulan dari komponen-komponen yang memiliki unsur keterkaitan antara satu dengan yang lainnya.

2.1.2 Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat-sifat tertentu, yaitu mempunyai komponen-komponen (*componen*), batas (*boundary*), lingkungan luar sistem (*environments*), penghubung (*interface*), masukan (*input*), keluaran (*output*), pengolah (*process*), dan sasaran (*objectives*) atau tujuan (*goal*).

a. Komponen–komponen sistem (*components*)

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerjasama membentuk suatu kesatuan.

Komponen–komponen sistem atau elemen – elemen sistem dapat berupa subsistem atau bagian–bagian dari sistem.

b. Batas sistem (*boundary*).

Merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipasang sebagai suatu kesatuan. Batas suatu sistem menunjukan ruang lingkup (*scope*) dari sistem tersebut.

c. Lingkungan luar (*environment*).

Merupakan apapun diluar batas dari sistem yang mempengaruhi operasi sistem yang dapat bersifat menguntungkan dan merugikan sistem tersebut.

d. Penghubung (*interface*).

Merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber – sumber daya mengalir dari suatu subsistem ke subsistem yang lain.

e. Masukan sistem

Merupakan energi yang dimasukkan kedalam sistem. Masukan dapat berupa peralatan (*maintenance input*) dan masukan sinyal (*signal input*). Maintenance input adalah energi yang diproses agar didapatkan keluaran. Sebagai contoh didalam sistem komputer, program adalah maintenance input yang digunakan untuk mengoperasikan komputernya sedangkan data adalah signal input utuk diolah menjadi informasi.

f. Keluaran sistem

Suatu sistem dapat mempunyai suatu bagian pengolah yang akan merubah masukan menjadi keluaran. Suatu sistem produksi akan mengolah masukan berupa bahan baku dan bahan–bahan yang lain menjadi keluaran berupa barang jadi. Sistem akuntansi akan mengolah transaksi menjadi keluaran keuangan dan laporan–laporan lain yang dibutuhkan oleh manajemen.

g. Sasaran sistem (*goal*).

Suatu sistem pasti mempunyai tujuan (*goal*) atau sasaran(*objectives*). Kalau sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan sekali, masukan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem.

2.1.3 Klasifikasi Sistem

Menurut Jogiyanto (2005), sistem dapat diklasifikasikan dari beberapa sudut pandangan, diantaranya adalah sebagai berikut ini :

1. Sistem diklasifikasikan berdasarkan sebagai sistem abstrak (*abstract system*) dan sistem fisik (*physical system*).

   Sistem abstrak adalah sistem yang berupa pemikiran atau ide-ide yang tidak nampak, misalnya sistem teologi. Sistem fisik adalah sistem yang ada secara fisik, misalnya sistem komputer.

2. Sistem diklasifikasikan sebagai sistem alamiah (*natural system*) dan sistem buatan manusia.

   Sistem alamiah adalah sistem yang terjadi melalui proses alam dan tidak dibuat manusia. Misalnya sistem perputaran bumi. Sistem buatan

manusia adalah sistem yang dirancang oleh manusia yang melibatkan interaksi manusia dengan mesin yang disebut dengan *human-machine system* atau *man-machine system*.

3. Sistem diklasifikasikan sebagai sistem tertentu (*deterministic system*) dan sistem tak tentu.

   Sistem tertentu beroperasi dengan tingkah laku yang sudah dapat diprediksi. Interaksi diantara bagian-bagiannya dideteksi dengan pasti, sehingga keluaran dari sistem dapat diramalkan. Misalnya sistem pada komputer. Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksikan karena mengandung unsur probabilitas.

4. Sistem diklasifikasikan sebagai sistem tertutup (*closed sistem*) dan sistem terbuka (*open system*).

   Sistem tertutup adalah sistem yang tidak berhubungan dan tidak terpengaruh dengan lingkungan luarnya. Sistem ini bekerja secara otomatis tanpa adanya turut campur tangan dari pihak luarnya. Secara teoritis, sistem tertutup ini ada, tetapi kenyataannya tidak ada sistem yang benar-benar tertutup. Yang ada hanyalah *relatively closed system* (secara relatif tertutup, tidak benar-benar tertutup). Sistem terbuka adalah sistem yang berhubungan dan terpengaruh dengan lingkungan luar atau subsistem yang lainnya. Karena sistem sifatnya terbuka dan terpengaruh oleh lingkungan luarnya, maka suatu sistem harus mempunyai suatu sistem pengendalian yang baik. Sistem yang baik harus dirancang sedemikian rupa sehingga secara relatif tertutup karena

sistem tertutup akan secara otomatis terbuka hanya untuk pengaruh yang baik.

2.1.4 Pengertian Informasi

Menurut Kadir (2008), informasi adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan saat ini atau saat mendatang.

2.1.5 Pengertian Sistem Informasi

Sistem Informasi merupakan kumpulan dari perangkat keras dan perangkat lunak komputer serta perangkat manusia yang akan mengolah data menggunakan perangkat keras dan perangkat lunak tersebut, selain itu data juga memegang peranan yang penting dalam sistem informasi, data yang akan dimasukkan dalam sebuah sistem informasi dapat berupa formulir-formulir, prosedur-prosedur, dan bentuk data lainnya.

Menurut Kadir (2008), dalam suatu sistem informasi terdapat komponen-komponen seperti :

a. Perangkat keras *(hardware)*: mencakup piranti-piranti fisik seperti komputer dan printer.
b. Perangkat lunak *(software)* atau program: sekumpulan instruksi yang memungkinkan perangkat keras untuk dapat memproses data.
c. Prosedur: sekumpulan aturan yang dipakai untuk mewujudkan pemrosesan data dan pembangkitan keluaran yang dikehendaki.

d. Orang: semua pihak yang bertanggung jawab dalam pengembangan sistem informasi, pemrosesan, dan penggunaan keluaran sistem informasi.
e. Basis data (*database*): sekumpulan tabel, hubungan, dan lain-lain yang berkaitan dengan penyimpanan data.
f. Jaringan komputer dan komunikasi data: sistem penghubung yang memungkinkan sumber (*resources*) dipakai secara bersama atau diakses oleh sejumlah pemakai.

2.2 Pengertian Rancang Bangun Sistem

Menurut Pressman (2012), perancangan adalah langkah pertama dalam fase pengembangan rekayasa produk atau sistem. Perancangan itu adalah proses penerapan berbagai teknik dan prinsip yang bertujuan untuk mendefinisikan sebuah peralatan, satu proses atau satu sistem secara detail yang membolehkan dilakukan realisasi fisik. Menurut Whiten dkk (2007), bangun sistem adalah membangun sistem informasi dan komponen yang didasarkan pada spesifikasi desain.

Dengan demikian pengertian rancang bangun merupakan kegiatan menerjemahkan hasil analisis ke dalam bentuk paket perangkat lunak kemudian menciptakan sistem tersebut ataupun memperbaiki sistem yang sudah ada.

2.3 Suku Cadang

Menurut Richardus Eko Indrajit dan Rhicardus Djokopranoto dalam bukunya Manajemen Persediaan menyatakan definisi suku cadang adalah sebagai berikut : “Suku cadang atau sparepart adalah suatu alat yang

mendukung pengadaan barang untuk keperluan peralatan yang digunakan dalam proses produksi".

Berdasarkan definisi diatas, suku cadang merupakan faktor utama yang menentukan jalannya proses produksi dalam suatu perusahaan. Sehingga dapat dikatakan suku cadang ini mempunyai peranan yang cukup besar dalam serangkaian aktivitas perusahaan.

2.4 Metode Pengembangan

2.4.1 Metode *Agile*

Konsep *Agile Software Development* dicetuskan oleh Kent Beck dan 16 rekannya dengan menyatakan bahwa *agile software development* adalah cara membangun software dengan melakukannya dan membantu orang lain sekaligus. Menurut Pressman (2010), *Agile software development methods* atau *agile methodology* merupakan sekumpulan metodologi pengembangan perangkat lunak yang berbasis pada pengembangan iterative, dimana persyaratan dan solusi berkembang melalui kolaborasi antar tim yang terorganisir. Sementara Sommerville (2011) mengemukakan metode a*gile* merupakan metode pengembangan *incremental* yang focus pada perkembangan yang cepat, perangkat lunak yang dirilis bertahap, mengurangi *overhead* proses, dan menghasilkan kode berkualitas tinggi dan pada proses perkembangannya melibatkan pelanggan secara langsung.

Ada beberapa lamgkah dalam *Agile Development Methods*, yaitu :

1. Perencanaan

   Pada langkah ini pengembang dan klien membuat rencana mengenai kebutuhan dari perangkat lunak yang akan dibuat.

2. Implementasi

   Bagian dari proses dimana programmer melakukan pengkodean perangkat lunak.

3. Tes Perangkat Lunak

   Disini perangkat lunak yang telah dibuat di tes oleh bagian control kualitas agar bug yang ditemukan bisa segera diperbaiki dan kualitas perangkat lunak terjaga.

4. Dokumentasi

   Setelah dilakukan tes perangkat lunak langkah selanjutnya yaitu proses dokumentasi perangkat lunak untuk mempermudah proses maintenance kedepannya.

5. Deployment

   Proses yang dilakukan oleh penjamin kualitas untuk menguji kualisa sistem. Setelah sistem memenuhi syarat maka perangkat lunak siap di deployment.

6. Pemeliharaan

   Langkah terakhir yaitu pemeliharaan. Tidak ada perangkat lunak yang 100% terbebas dari bug, oleh karena itu sangatlah penting agar perangkat lunak dipelihara secara berkala.

2.5 Konsep Dasar Web

2.5.1 *Website*

Menurut Murad, dkk (2013), *Website* adalah sistem dengan informasi yang disajikan dalam bentuk teks, gambar, suara, dan lainnya

yang tersimpan dalam sebuah s*erver web* internet yang disajikan dalam bentuk *hypertext*. Dapat disimpulkan bahwa *website* adalah sebuah tempat di internet yang menyajikan informasi dengan berbagai macam format data seperti teks, gambar, bahkan video yang dapat diakses menggunakan berbagai aplikasi klien sehingga memungkinkan penyajian informasi yang lebih menarik dan dinamis dengan pengelolaan yang terorganisasi. *Website* memiliki teknologi yang dikenal sebagai *web browser*, *web hosting*, dan *web server*.

2.5.2 *Web Server*

Menurut Anhar (2010), *Web server* adalah aplikasi yang berfungsi untuk melayani permintaan pemanggilan alamat dari pengguna melalui *web browser*, dimana *web server* mengirimkan kembali informasi yang diminta tersebut melalui HTTP (*Hypertext Transfer Protocol*) untuk ditampilkan ke layar monitor komputer kita. Agar kita dapat mengubah isi dari *website* yang dibuat, kita membutuhkan program PHP. *Script-Script* PHP tersebut yang berfungsi membuat *website* halaman menjadi dinamis. Dinamis artinya pengunjung *web* dapat memberikan komentar dan saran pada *website* kita.

2.6 Perangkat Lunak Yang Digunakan

2.6.1 XAMPP

Menurut Madcoms (2011) sekarang ini banyak paket *software* instalasi *web server* yang disediakan secara gratis diantaranya menggunakan XAMPP. Dengan menggunakan paket *software* instalasi ini,

maka sudah dapat melakukan beberapa instalasi *software* pendukung *web server*, yaitu *Apache*, *PHP*, *phpMyAdmin*, dan *database* MySQL. Fungsinya adalah sebagai *server* yang berdiri sendiri (*localhost*), yang terdiri atas program *Apache HTTP Server*, MySQL *database*, dan penerjemah bahasa yang ditulis dengan bahasa pemrograman PHP dan *Perl*. Nama XAMPP merupakan singkatan dari X (empat sistem operasi apapun), *Apache*, MySQL, PHP, dan *Perl*. Program ini tersedia dalam GNU *General Public License* dan bebas digunakan untuk umum.

2.6.2 PHP

Menurut Madcoms (2011), PHP adalah pemrograman interpreter yaitu proses penerjemahan baris kode mesin yang dimengerti komputer secara langsung pada saat baris kode dijalankan atau sering disebut suatu bahasa dengan hak cipta terbuka atau yang juga dikenal dengan istilah *open source* yaitu pengguna dapat mengembangkan kode-kode fungsi PHP sesuai dengan kebutuhannya. Dapat ditarik kesimpulan bahwa PHP adalah bahasa pemrograman yang digunakan secara luas untuk menangani pembuatan dan pengembangan sebuah situs *web* dan bisa digunakan bersamaan dengan HTML.

2.6.3 *Sublime Text*

Menurut Supono dan Putratama (2016), *Sublime Text* merupakan perangat lunak *text editor* yang digunakan untuk membuat atau mengedit suatu aplikasi. *Sublime text* mempunyai fitur plugin tambahan yang memudahkan programmer.

2.6.4 HTML (*Hypertext Markup Lanuage*)

Menurut Anhar (2010), HTML (*Hypertext Markup Language*) adalah sekumpulan simbol-simbol atau *tag-tag* yang dituliskan dalam sebuah *file* yang digunakan untuk menampilkan halaman pada *web browser*. Pada dokumen HTML yang termasuk sistem *hypertext*, kita tidak harus membaca dokumen tersebut secara urut dari atas ke bawah atau sebaliknya, tetapi kita dapat menuju topik tertentu secara langsung dengan menggunakan teks penghubung yang akan membawa anda ke suatu topik atau dokumen lain secara langsung. Dokumen ini umumnya berisi informasi atau *interface* aplikasi di dalam internet. Ada dua cara untuk membuat sebuah *web page* yaitu dengan HTML *editor* atau dengan *editor* teks biasa seperti *notepad*.

2.6.5 Basis Data

Menurut Indrajani (2015), basis data adalah kumpulan data yang saling berhubungan secara logis dan didesain untuk mendapatkan data yang dibutuhkan oleh suatu organisasi.

2.6.6 MySQL

Menurut Anhar (2010), MySQL (*My Structure Query Language*) adalah sebuah perangkat lunak sistem manajemen basis data SQL (*Database Management System*) atau DBMS. Dapat disimpulkan MySQL adalah salah satu jenis *database server* yang termasuk jenis RDBMS (*Relational Database Management System*).

2.7 Peralatan Pendukung (*Tools System*)

2.7.1 Pengertian UML

Menurut Munawar (2005), *Unified Modeling Language* (UML) adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi obyek. Hal ini di sebabkan karena UML menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah di mengerti, serta di lengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

UML merupakan kesatuan dari bahasa pemodelan yang di kembangkan oleh Booch, *Object Modeling Technique* (OMT) dan *Object Oriented Software Engineering* (OOSE). Metode Booch dari Grady Booch sangat terkenal dengan nama metode *Design Object Oriented*. Metode ini menjadikan proses analisis dan *design* ke dalam 4 (empat) tahapan iterative, yaitu: identifikasi kelas-kelas dan obyek-obyek, identifikasi semantik dari hubungan obyek dan kelas tersebut, perincian *interface* dan implementasi. Keunggulan metode Booch adalah pada detil dan kayanya dengan notasi dan elemen. Pemodelan OMT yang di kembangkan oleh Rumbaugh di dasarkan pada analisis terstruktur dan pemodelan *entity-relationship.*

2.7.2 Use Case Diagram

Menurut Munawar (2005), *Use case* adalah deskripsi fungsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem

dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem di pakai.

Dalam sebuah pembicaraan tentang *use case*, pengguna biasanya di sebut dengan *actor*. *Actor* adalah sebuah peran yang bisa di mainkan oleh pengguna dalam interaksinya dengan sistem. *Use case* adalah alat bantu terbaik guna menstimulasi pengguna potensial untuk mengatakan tentang suatu sistem dari sudut pandangnya. Menurut Munawar (2005), diagram *use case* mempunyai 3 notasi yang menunjukkan aspek dari sistem :

a. *Actor* (Pengguna) yaitu abstraksi dari orang dan sistem lain yang mengaktifkan fungsi dari target sistem. *Actor* mewakili peran orang, sistem yang lain atau alat ketika berkomunikasi dengan *use case*.

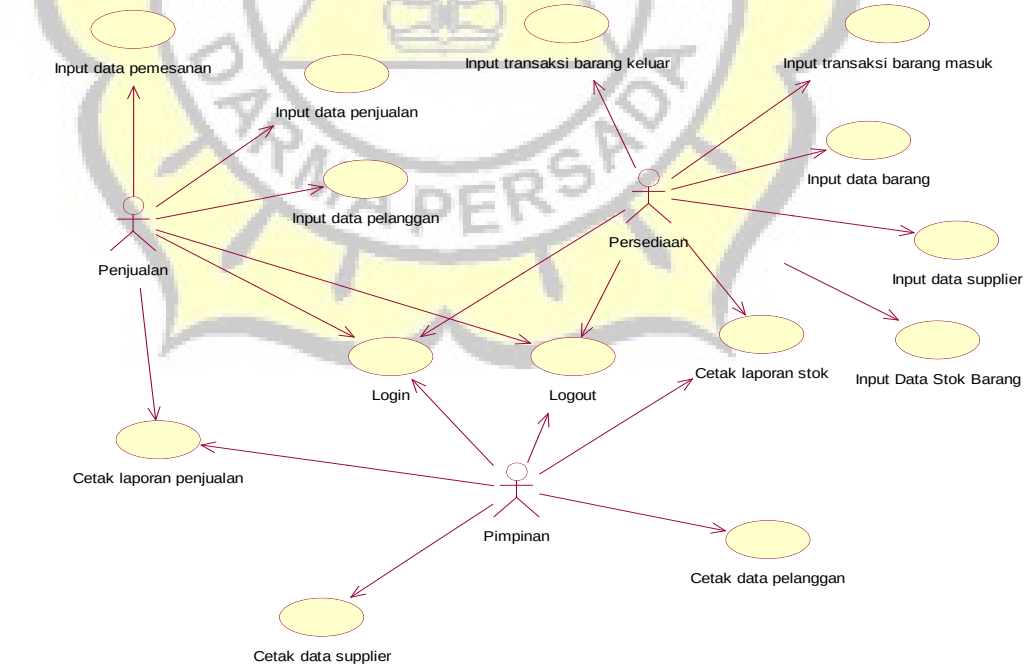
b. *Use Case* adalah abstraksi dari interaksi antara sistem dan *actor*. *Use case* di buat berdasarkan keperluan *actor*. *Use Case* harus merupakan "apa" yang di kerjakan *software* aplikasi, bukan "bagaimana" *software* aplikasi mengerjakannya. Setiap *use case* harus di beri nama yang menyatakan apa hal yang di capai dari hasil interaksinya dengan *actor*.

c. *Relationship* (hubungan) yaitu hubungan antara *actor*/pelaku dengan *use case* di mana terjadi interaksi di antara mereka.

Daftar notasi yang ada pada *use case diagram* adalah sebagai berikut.

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri (*independent*).
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *use case* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.
7		*System*	Menspesifikasikan paket yang menampilkan sistem secara terbatas.
8		*Use Case*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor

Gambar 2.1 Daftar Notasi *Use Case Diagram*

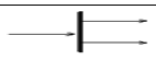
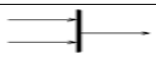


Gambar 2.2 Contoh Diagram Model *Use Case*

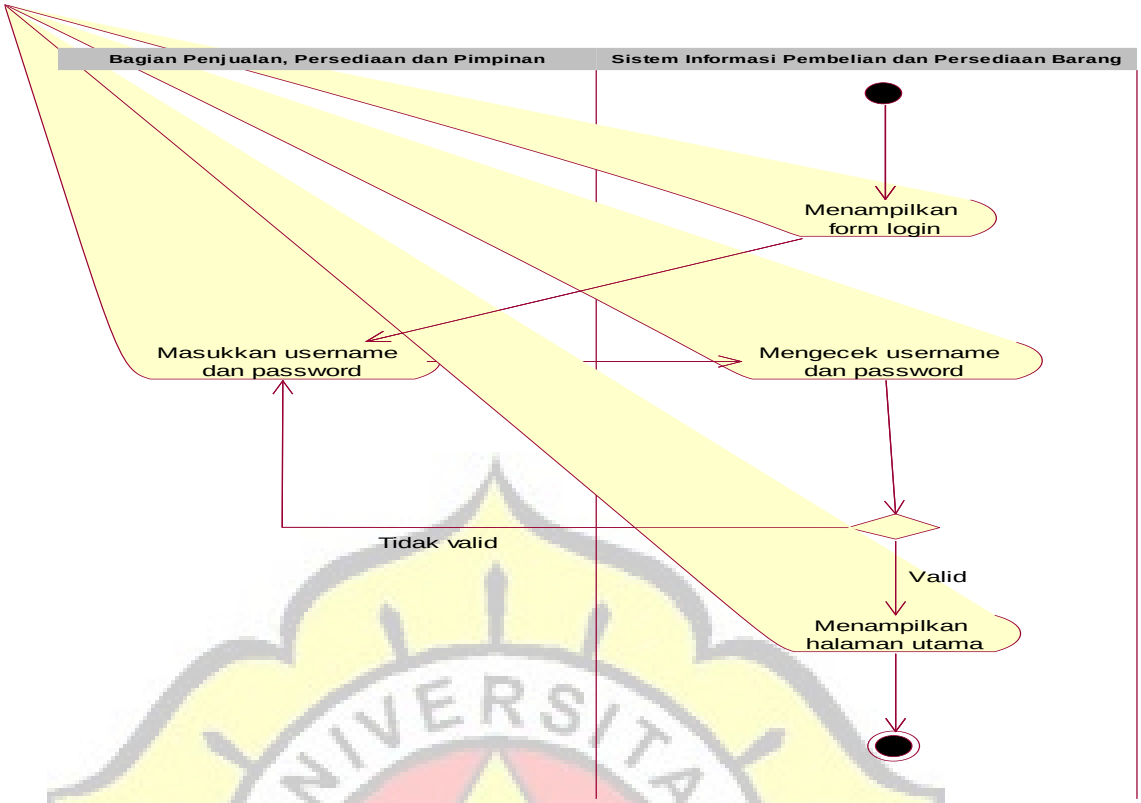
2.7.3 Activity Diagram

Menurut Munawar (2005), *Activity Diagram* adalah teknik untuk mendeskripsikan logika *procedural*, proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku paralel sedangkan *flowchart* tidak bisa.

Daftar notasi yang ada pada *activity diagram* adalah sebagai berikut.

NO	GAMBAR	NAMA	KETERANGAN
1		*Activity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain
2		*Action*	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		*Start State*	Bagaimana objek dibentuk atau diawali
4		*End State*	Bagaimana objek dibentuk dan dihancurkan
5		*State Transitio*	State transition menunjukkan kegiatan apa berikutnya setelah suatu kegiatan
6		*Fork*	Percabangan yang menunjukkan aliran pada *Activity Diagram*
7		*Join*	Penggabungan yang menjadi arah aliran pada *Activity Diagram*
8			Pilihan untuk mengambil keputusan
9		*Flow Final*	Aliran akhir

Gambar 2.3 Daftar Notasi *Activity Diagram*



Gambar 2.4 Contoh Model *Activity Diagram*

2.7.4 Skenario

Menurut Munawar (2010), scenario adalah sebuah dokumentasi terhadap kebutuhan fungsional dari sebuah sistem. Form scenario merupakan penjelasan penulisan *use case* dari sudut pandang a*ctor*.