BAB II
LANDASAN TEORI

2.1 Sistem

Menurut Rudy tantra (2012) dalam bukunya berjudul menajemen proyek system informasi, sistem adalah entitas atau satuan yang terdiri dari dua atau lebih komponen atau subsistem (sistem yang lebih kecil) yang saling terhhubung atau terkait untuk mencapai suatu tujuan.

Definisi Sistem menurut Abdul Kadir (2003) adalah sekumpulan elemen yang saling terkait atau terpadu yang dimaksudkan untuk mencapai suatu tujuan.

Tata Sutabri (2012) mendefinisikan Sistem adalah sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel yang terorganisir, saling berinteraksi, saling tergantung satu sama lain, dan terpadu.

Menurut Tata Sutabri (2012) sistem memiliki karakteristik atau sifat-sifat tertentu, yaitu :

1. Komponen sistem *(Component)*

   Susatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, artinya saling bekerja sama membentuk satu kesatuan. Komponen-komponen sistem tersebut dapat berupa suatu bentuk subsistem. Setiap subsistem memiliki sifat dari sistem yang menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai sistem yang lebih besar atau sering disebut “supra sistem”.
2. Batasan sistem *(Boundary)*

   Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem yang lain atau sistem dengan lingkung luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan yang tidak dapat dipisahkan.
3. Lingkungan luar sistem *(Envirotment)*

   Bentuk apapun yang ada diluar ruang lingkup atau batasan sistem yang memepengaruhi oprasi sistem tersebut disebut lingkungan luar sistem. lingkungan luar sistem ini dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut. Dengan demikian, lingkungan luar tersebut

harus tetap dijaga dan dipelihara. Lingkungan luar yang merugikan harus dikendalikan. Kalau tidak, maka akan mengganggu kelangsungan hidup sistem tersebut.

4. Penghubung sistem *(Interface)*

Media yang menghubungkan sistem dengan subsistem lain disebut penghubung sistem atau interface. Penghubung ini memungkinkan sumber-sember daya mengalir dari satu subsistem ke subsistem lain. Bentuk keluaran dari subsistem akan menjadi masukan untuk subsistem lain melalui penghubung tersebut. Dengan demikian, dapat terjadi suatu integrasi sistem yang membentuk satu kesatuan.

5. Masukan sistem *(Input)*

Energi yang dimasukkan kedalam sistem tersebut masukkan sistem, yang dapat berupa pemeliharaan *(maintenance input)* dan sinyal *(signal input).*

Contoh, di dalam suatu unit sistem komputer, "Program" adalah *maintenance* input yang digunakan untuk mengoprasikan komputernya dan "Data" adalah signal input untuk diolah menjadi informasi.

6. Keluaran sistem *(Output)*

Hasil energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna. Keluaran ini merupakan masukan bagi subsistem yang lain seperti sistem informasi. Keluaran yang dihasilkan adalah informasi. Informasi ini dapat digunakan sebagai masukan untuk pengambilan keputusan atau hal-hal lain yang menjadi input bagi subsistem lain.

7. Pengolahan sistem *(Proses)*

Suatu sistem dapat mempunyai susatu proses yang akan mengubah masukan menjadi keluaran, contohnya adalah sistem akuntansi. Sistem ini akan mengolah data transaksi menjadi laporan-laporan yang dibutuhkan oleh pihak manajemen.

8. Sasaran sistem *(Objective)*

Suatu sistem memiliki tujuan dan sasaran yang pasti dan bersifat *deterministic*. Kalau suatu sistem tidak memiliki sasaran maka oprasi sistem tidak ada gunanya. Suatu sistem dikatakan berhasil bilai mengenai sasaran dan tujuan yang telah direncanakan.

2.2 Perancangan

Dalam buku berjudul rekayasa perangkat lunak pengarang Roger S.pressman, Ph.D, Jacob Nielsen (nie00) mengatakan: “pada dasarnya ada 2 pendekatan yang bersifat mendasar pada perancangan perangkat lunak :idealisme arsitek yang mengekspresikan diri anda sendiri dan idealism rekayasa yang bertujuan untuk menyelesaikan permasalahan-permasalahan yang ada di arah pelanggan “ selama dekade pertama pengembangan web ,idealisme arsitek merupakan pendekatan yang paling banyak dipilih oleh para pengembang aplikasi web. Perancanggan dilakukan dengan prinsip ini dan biasannya dilakukan saat HTML *(hypertext markup language*) dibentuk. Perancangan aplikasi web menekankan pada visi artistic yang kemudian akan berkembang saat konstruksi aplikasi web dilakukan.

2.3 E-Penjualan (penjualan elektronik)

Penjualan elektronik atau *e-commerce* adalah segala bentuk kegiatan pembelian dan penjualan,pemasaran produk,jasa,dan informasi yang dilakukan secara elektronis. Domain e-*commerce* berupa B2B, B2C, dan C2C (kadir,abdul, dkk.,pengantar tekhnologi informasi,2013,hal.371)

2.4 Internet

internet merupakan contoh sebuah jaringan komputer. Jaringan ini menghubungkan jutaan komputer yang tersebar diseluruh dunia dan siapa pun dapat terhubung ke jaringan ini. Internet banyak memberikan keuntungan pada pemakai,namun dibalik manfaat yang bisa diperoleh internet juga membawa dampak negatif. (kadir,abdul, dkk., pengantar tekhnologi informasi ,2013,hal.300)

2.5 Perangkat Lunak yang Digunakan

1. Visual Studio

Microsoft Visual Studio merupakan sebuah perangkat lunak lengkap (suite) yang dapat digunakan untuk melakukan pengembangan aplikasi. Baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya dalam bentuk aplikasi console, aplikasi Windows,ataupun aplikasi Web.Visual Studio mencakup kompiler, SDK,

Integrated Development Environment (IDE), dan dokumentasi (umumnya berupa MSDN Library). Kompiler yang dimasukkan kedalam paket Visual Studio antara lain *Visual C++*, Visual C#, *Visual Basic*, *Visual Basic.NET*, *Visual InterDev*, Visual J++, Visual J#, Visual FoxPro, dan *Visual Source Safe*.

Microsoft Visual Studio dapat digunakan untuk mengembangkan aplikasi dalam *native code* (dalam bentuk bahasa mesin yang berjalan di Windows) ataupun managed code (dalam bentuk *Microsoft Intermediate Language* di atas. *NET Framework*). Selain itu, Visual Studio juga dapat digunakan untuk mengembangkan aplikasi *Silverlight,* aplikasi *Windows Mobile* (yang berjalan di atas . *NET Compact Framework*).

2. PHP

PHP berasal dari kata *hypertext preprocessor*, yaitu bahasa pemrograman universal untuk penanganan pembuatan dan pengembangan sebuah situs web dan bisa digunakan bersamaan dengan HTML. PHP banyak digunakan untuk membuat situs web dinamis .

3. MySQL

MySQL adalah system menajemen *database* SQL yang sifatnya *open source* (terbuka) dan paling banyak digunakan saat ini. System *database* MySQL mampu mendukung beberapa fitur seperti *multithread*, *multi-user*, dan SQL *database menagemen system* (DBMS).

2.6 Database

Basis data *(Database)* adalah suatu pengorganisasian sekumpulan data yang saling terkait sehingga memudahkan aktivitas untuk memperoleh informasi. (kadir,abdul, dkk., pengantar tekhnologi informasi, 2013, hal.339). Secara umum, Database berarti koleksi data yang saling terkait. Secara praktis, basis data dapat dianggap sebagai satu penyusunan data yang terstruktur yang disimpan dalam media pengingat (*hard disk*) yang tujuannya adalah agar data tersebut dapat diakses dengan mudah dan cepat.

Beberapa pengertian Database oleh beberapa ahli:

- Menurut CJ.Date

  Database adalah koleksi data operasional yang tersimpan dan juga dipakai oleh system aplikasi dari suatu organisasi.

  - Data *input* ialah data yang masuk dari luar sistem
  - Data *output* ialah data yang di hasilkan oleh sistem
  - Data operasional ialah data yang tersimpan pada sistem

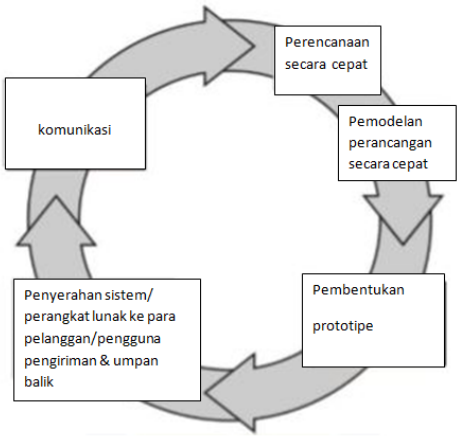
- Menurut Gordon C.Everest

  Database adalah koleksi atau kumpulan data yang mekanis, terbagi, terdefinisi secara formal dan juga dikontrol terpusat pada suatu organisasi.

2.7 Metode pengembangan perangkat lunak

2.7.1 Metode *Prototype*

Pressman (2012:50) menjelaskan, dalam melakukan perancangan sistem yang akan dikembangkan dapat mengunakan metode prototype. Metode ini cocok digunakan untuk mengembangkan sebuah perangkat yang akan dikembangkan kembali. Metode ini dimulai dengan pengumpulan kebutuhan pengguna, dalam hal ini pengguna dari perangkat yang dikembangkan adalah peserta didik. Kemudian membuat sebuah rancangan kilat yang selanjutnya akan dievaluasi kembali sebelum diproduksi secara benar.



Gambar 2.1 Paradigma pembuatan prototype

Pembuatan prototipe (gambar 1.) dimulai dengan dilakukanya komunikasi antara tim pengembang perangkat lunak dengan pada pelanggan. Tim pengembang perangkat lunak akan melakukan pertemuan – pertemuan dengan para *stakeholder* untuk mendefinisikan sasaran keseluruhan untuk perangkat lunak yang akan dikembangkan, mengindentifikasikan spesifikasi kebutuhan apa pun yang saat ini di ketahui, dan menggambarkan area – area dimana definisi lebih jauh pada iterasi selanjutnya merupakan keharusan. Iterasi pembuatan prototipe direncanakan dengan cepat dan pemodelan (dalam bentuk " rancangan cepat).

2.9 *Unified Modeling Language* (UML)

UML(*unified modeling language*) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk medefinisikan requirement, membuat analisis dan desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek.

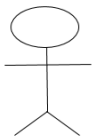
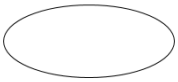
Menurut Wikipedia, UML adalah bahasa spesifikasi standar untuk mendokumentasikan, menspesifikasikan dan membangun sistem software. UML adalah hasil pengembangan dari bahasa pemodelan berorientasi objek (*object oriented modelling language).* Saat ini, versi UML yang digunakan adalah versi 2.0 yang merupakan hasil dari pengembangan metode yang dikreasikan oleh grady booch, jim raumbaugh, dan ivar Jacobson. tipe-tipe diagaram UML adalah sebagai berikut :

2.9.1 Use Case Diagram

Use case diagaram adalah gambar dari beberapa atau seluruh aktor dan *use case* dengan tujuan yang mengenali interaksi mereka dalam suatu sistem. *Use case diagram* menggambarkan fungsionalitas yang diharapkan dari sebuah sistem, yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* mepresentasikan sebuah interaksi antara actor dan sistem.

Dalam *use case* diagram terdapat istilah seperti aktor, *use case* dan *case relationship.* Penjelasan simbol pada tabel 2.1

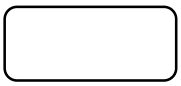
Tabel 2.1 Simbol Use Case

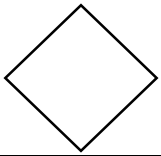
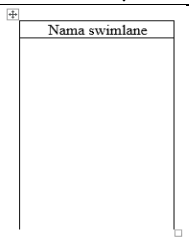
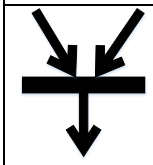
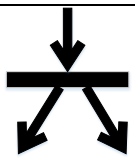
Simbol	Keterangan
	Aktor : Seseorang atau sesuatu yang berinteraksi dengan sistem yang sedang dikembangkan.
	*Use case* : perungkat tertinggi dari fungsionalitas yang dimiliki sistem.
	Association :adalah relasi antara actor dan *use case.*
	Generalisasi : untuk memperlihatkan struktur pewaris yang terjadi.

2.9.2 Activity Diagram

Activity diagram menggambarkan rangkaian aliran dari aktifitas, digunakan untuk mendeskripsikan aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi. *Activity Diagram* berupa flow chart yang digunakan untuk memperlihatkan aliran kerja dari sistem. Notasi yang digunakan dalam *activity diagram* adalah sebagai berikut :

Tabel 2.2 Simbol *Activity Diagram*

Simbol	**Keterangan**
	*Activity* : Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
	*Initial Node* : Bagaimana objek dibentuk atau diawali
	*Actifity Final Node* : Bagaimana objek dibentuk dan diakhiri.

	*Decision* : Asosiasi percabangan dimana jika ada pilihan aktifitas lebih dari satu.
	*Swimlane* : Memisalkan organisasi bisnis yang bertanggung jawab terhadap aktifitas yang terjadi.
	*Join* : Digunakan untuk menunjukkan kegiatan yang digabungkan.
	*Fork* : Digunakan untuk menunjukkan kegiatan yang dilakukan secara paralel

2.9.3 *Sequence Diagram*

Sequence diagram menggambarkan kolaborasi dinamis antara sejumalah dan untuk menunjukkan rangkaian pesan yang dikirim antar objek juga interaksi antar objek, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem. *Sequence* diagram menjelaskan interaksi objek yang disusun berdasarkan urutan waktu.

Dalam *sequence* diagram terdapat 2 simbol yaitu :

a. Actor, untuk menggambarkan pengguna sistem.
b. *Lifeline,* untuk menggambarkan kelas dan objek.

2.9.4 Class Diagram

Class diagram menggambarkan dtruktur data dan desripsi *class, package, dan* objek beserta hubungan satu sama lain. *Class* diagram berfungsi untuk menjelaskan tipe dari objek sistem dan hubungannya dengan objek yang lain. *Class* memiliki 3 area pokok yaitu nama, atribut dan metode.

2.10 *White Box Testing*

White box adalah meramalkan cara kerja perangkat lunak secara rinci, kerenanya *logical path* (jalur logika) perangkat lunak akan di test, denan menyediakan *test case* yang akan mengerjakan kumpulan kondisi dan atau pengulangan secara spesifik. Secara sekilas dapat diambil kesimpulan *white box* merupakan petunjuk untuk mendapatkan program yang benar secara 100%.

Uji coba *white box*:

Uji coba ini adalah metode perancangan *test case* yang meggunakan struktur kontrol dari perancangan prosedural untuk mendapatkan *test case*. Dengan menggunakan metode *white box*, analisis system akan mendapat *test case* yang:

- Menjamin seluruh *independent path* didalam modul yang dikerjakan sekurang-kurangnya sekali
- Mengerjakan seluruh keputusan *logical*
- Mengerjakan seluruh *loop* yang sesuai dengan batasannya
- Mengerjakan seluruh struktur data internal yang menjamin validasi

2.11*Black Box Testing*

Test case ini bertujuan untuk menunjukan fungsi PL tentang cara beroperasinya, apakah pemasukan data keluaran telah berjalan sebagaimana yang diharapkan dan apakah informasi yang disimpan secara eksternal selalu dijaga kemutakhirannya.

Pengujian *black box testing*:

Pengujian *black box* berfokus pada persyaratan funsional PL, pengujian ini memungkinkan analis sistem memperoleh kumpulan kondisi input yang akan mengerjakan seluruh keperluan fungsional program.

Tujuan metode ini mencari kesalahan pada:

- fungsi yang salah atau hilang
- kesalahan pada *interface*
- kesalahan pada struktur data atau akses *database*
- kesalahan performansi
- kesalahan inisialisasi dan tujuan akhir