

BAB II

LANDASAN TEORI

2.1 Tinjauan Pustaka

Menurut Arman Suryadi Karim (2016) dalam jurnal Informatika – Volume 2 no. 2 (2016). Yang berjudul Sistem Informasi Tugas Akhir *Online* Berbasis Web sistem informasi berbasis web ini, membantu dalam pendaftaran Tugas akhir, dan mempermudah dalam laporan data. Dalam penelitian yang berjudul “ Sistem Informasi Tugas Akhir *Online* Berbasis Web” berharap dapat mengembangkan hasil peneliti terdahulu, selanjutnya mampu memberikan kontribusi dan layanan yang lebih baik.

Menurut Teguh Cahyono (2011) dalam jurnal Informatika – Volume 3 no. 22 (2011). Yang berjudul Perancangan Sistem Informasi Pengelolaan Tugas Akhir mereka mengatakan sistem yang berjalan belum optimal dikarenakan masih manual, yaitu masih menggunakan pembukuan. Oleh karena itu, karena itu dalam mengelola data mahasiswa belum cukup efektif dan efisien, dengan membangun sistem yang terkomputerisasi dan berbasis web, maka dapat mempermudah pihak akademik dalam mengelola data mahasiswa.

Menurut Ahmad Muttaqin (2008) dalam jurnal Informatika – Volume 4 no. 5 (2008). Yang berjudul Sistem Informasi Pengelolaan Tugas Akhir Berbasis Web dengan PHP dan MySQL dalam sistem yang berjalan pada akademik masih kurang optimal, dalam pendaftaran tugas akhir masih menggunakan form kertas, dan memperlambat pihak akademik dalam pembuatan laporan, dengan dikembangkannya sistem ini menjadi terkomputerisasi, diharapkan dapat membantu pihak akademik dan mahasiswa dalam mendaftarkan tugas akhir.

1.2 Pengertian Sistem

Sistem pada dasarnya adalah sekelompok unsur yang erat hubungannya satu dengan yang lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu. Secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel-variabel yang terorganisasi, saling berinteraksi, saling

tergantung satu sama lain dan terpadu. Sistem bisa berupa abstraksi atau fisis (Gordon B. Davis, 2002). Sistem yang abstrak adalah susunan yang teratur dari gagasan-gagasan atau konsepsi yang saling tergantung. Sedangkan sistem yang bersifat fisis adalah serangkaian unsur yang bekerjasama untuk mencapai suatu tujuan (Tata Sutabri, 2004).

Dari definisi di atas maka dapat diketahui manfaat sistem yaitu untuk menyatukan atau mengintegrasikan semua unsur yang ada dalam suatu ruang lingkup, dimana komponen-komponen tersebut tidak dapat berdiri sendiri. Komponen atau sub sistem harus saling berintegrasi dan saling berhubungan untuk membentuk satu kesatuan sehingga sasaran dan tujuan dari sistem tersebut dapat tercapai. Pendekatan sistem yang merupakan kumpulan dari komponen atau elemen-elemen merupakan definisi yang lebih luas dibandingkan dengan pendekatan sistem yang *prosedural*.

Definisi lain dari sistem adalah kumpulan dari bagian-bagian yang bekerja sama untuk mencapai tujuan yang sama atau sekumpulan objek-objek yang saling berelasi dan berinteraksi (Hanif Al Fata, 2007).

2.2.1 Komponen Dasar Sistem

Suatu sistem mempunyai karakteristik atau sifat-sifat yang tertentu yaitu mempunyai komponen-komponen (*components*), batas sistem (*boundary*), lingkungan luar sistem (*environments*), penghubung (*interface*), masukan (*input*), keluaran (*output*), pengolah (*process*), dan sasaran (*objectives*) atau tujuan (*goal*). Jogyanto (2008 : 3).

Karakteristik sistem mempunyai beberapa komponen diantaranya yaitu :

a. **Komponen sistem (*component*)**

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk suatu kesatuan. Komponen-komponen sistem atau elemen-elemen sistem dapat berupa suatu subsistem atau bagian-bagian dari sistem. Setiap sub-sub sistem mempunyai sifat-sifat dari sistem yang menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. (Jogyanto, 2008 : 4).

b. Batas sistem (*boundary*)

Batas sistem (*boundary*) merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lain atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan. Batas suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

c. Lingkungan luar sistem (*environment*)

Lingkungan luar (*environment*) dari suatu sistem adalah apapun diluar batas dari sistem dapat bersifat menguntungkan dan dapat juga merugikan. Lingkungan luar yang menguntungkan merupakan energi dari sistem dengan demikian harus dijaga dan dipelihara, sedangkan lingkungan luar yang merugikan harus ditahan dan dikendalikan kalau tidak maka akan mengganggu kelangsungan hidup dari sistem.

d. Penghubung (*interface*)

Penghubung (*interface*) merupakan media penghubung antara subsistem dengan subsistem lain. Melalui penghubung ini memungkinkan sumber-sumber daya mengalir dari satu subsistem ke subsistem lain. Keluaran (*output*) dari subsistem akan menjadi masukan (*input*) pada sistem lain dengan penghubung satu subsistem dapat berinteraksi dengan subsistem lain membentuk satu kesatuan. (Jogiyanto, 2008 : 5).

e. Masukan (*input*)

Masukan (*input*) sistem adalah energi yang dimasukkan kedalam sistem. Masukan dapat berupa masukan perawatan (*maintenance input*) dan masukan (*signal input*). *Maintenance input* adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. *Signal input* adalah energi yang diproses untuk didapatkan keluarannya.

f. Keluaran (*output*)

Merupakan hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran berguna dan sisa pembuangan. Keluaran dapat merupakan masukan untuk subsistem yang lain atau kepada supra sistem.

g. Pengolah (*process*)

Suatu sistem dapat mempunyai suatu bagian pengolahan atau sistem itu sendiri sebagai pengolahnya. Pengolah yang akan merubah masukan menjadi keluaran.

h. Sasaran (*objective*) atau tujuan (*goal*)

Suatu sistem pasti mempunyai tujuan (*goal*) atau sasaran (*objectives*). Kalau suatu sistem tidak mempunyai sasaran maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran aturan tertentu.

2.3 Pengertian Skripsi

Skripsi dapat diartikan sebagai karya tulis yang disusun oleh mahasiswa yang telah menyelesaikan kurang lebih 140 sks dengan bimbingan oleh dosen pembimbing sebagai salah satu persyaratan untuk mencapai gelar pendidikan S-1

Ada beberapa pengertian lain dari skripsi :

Skripsi merupakan karya tulis ilmiah berdasarkan hasil penelitian lapangan dan atau studi kepustakaan yang disusun mahasiswa sesuai dengan bidang studinya sebagai tugas akhir dalam studi formalnya diperguruan tinggi. Skripsi dalam dunia pendidikan berarti suatu hasil penyusunan tulisan ilmiah yang telah dibuktikan kebenarannya berdasarkan data-data yang telah dikumpulkan dan tentunya data yang dikumpulkan diolah untuk kemudian dijadikan data yang valid sebagai bahan acuan buat membuktikan kebenaran suatu tulisan tersebut.

Skripsi adalah laporan tertulis hasil penelitian yang dilakukan oleh mahasiswa dengan bimbingan dosen pembimbing skripsi untuk dipertahankan dihadapan dosen penguji skripsi sebagai syarat untuk memperoleh drajat sarjana skripsi

2.4 Unified Modeling Language

Unified Modeling Language (UML) adalah tujuan umum, perkembangan, bahasa pemodelan dibidang rekayasa perangkat lunak, yang dimaksudkan untuk menyediakan cara standar untuk memvisualisasikan desain sistem.

UML awalnya termotivasi oleh keinginan untuk membakukan sistem notasi yang berbeda dan pendekatan untuk desain perangkat lunak yang dikembangkan oleh Grady Booch, Ivar Jacobson dan James Rumbaugh di Rational Software pada 1994 – 1995, dengan pengembangan lebih lanjut yang dipimpin oleh mereka melalui tahun 1996.

Pada tahun 1997 UML diadopsi sebagai standar oleh Object Management Group (OMG), dan telah dikelola oleh organisasi ini. Pada tahun 2005 UML juga diterbitkan oleh International Organization for Standardization (ISO) sebagai standar ISO. Sejak itu telah periodik direvisi untuk menutupi revisi terbaru dari UML.

2.4.1 Use Case Diagram

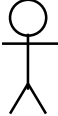
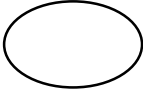

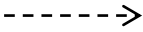


Use Case menggambarkan apa saja aktifitas yang dilakukan suatu sistem dari sudut pandang pengamatan luar, yang menjadi persoalan itu apa yang dilakukan bukan bagaimana melakukannya. *Use Case* dekat kaitannya dengan kejadian-kejadian. Kejadian (*scenario*) merupakan contoh apa yang terjadi ketika seseorang berinteraksi dengan sistem.

Use Case Diagram berguna dalam tiga hal :

1. Menjelaskan fasilitas yang ada (*requirement*)
Use Case baru selalu menghasilkan fasilitas baru ketika sistem dianalisa, dan *design* menjadi lebih jelas.
2. Komunikasi dengan klien
Penggunaan notasi dan simbol dalam diagram *Use Case* membuat pengembang lebih mudah berkomunikasi dengan klien-klienya.
3. Membuat test dari kasus-kasus secara umum
Kumpulan dari kejadian-kejadian untuk *Use Case* bisa dilakukan test kasus layak untuk kejadian-kejadian tersebut.

Tabel 2.1 Simbol-simbol *Use case* Diagram

| Nama | Simbol | Keterangan |
|------|--------|------------|
|------|--------|------------|

| | | |
|----------------------|---|--|
| <i>Actor</i> |  | Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan use case |
| <i>Use Case</i> |  | Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor |
| <i>Collaboration</i> |  | Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan perilaku yang lebih besar dari jumlah dan elemen-elemennya |
| <i>Include</i> |  | Menspesifikasikan bahwa use case sumber secara eksplisit |
| <i>Extend</i> |  | Menspesifikasikan bahwa use case target memperluas perilaku dari use case sumber pada suatu titik yang diberikan |
| <i>Association</i> |  | Menghubungkan antara objek satu dengan objek lainnya |

2.4.2 Activity Diagram


Pada dasarnya *Diagram Activity* sering digunakan oleh flowchart. Diagram ini berhubungan dengan diagram *StateChart*. *Diagram StateChart* berfokus pada objek yang dalam suatu proses (atau proses menjadi suatu objek), *Diagram Activity* berfokus pada aktifitas-aktifitas yang terkait dalam suatu proses tunggal. Jadi dengan kata lain, diagram ini menunjukkan bagaimana aktifitas-aktifitas tersebut bergantung satu sama lain.

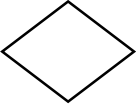
Diagram Activity dapat dibagi menjadi beberapa jalur kelompok yang menunjukkan objek yang mana yang bertanggung jawab untuk suatu aktifitas. Perlihatkan

tunggal (*single transision*) timbul dari setiap adanya *activity* (aktifitas), yang saling menghubungkan pada aktifitas berikutnya. Sebuah transision (transisi) dapat membuat cabang kedua atau lebih percabangan *exclusif transision* (transisi eksklusif).

Label *Guard Exprssion* (ada didalam []) yang menerangkan output (keluaran) dari percabangan. Percabangan akan menghasilkan bentuk menyerupai bentuk instan. Transisi bisa bercabang menjadi beberapa aktifitas paralel yang disebut *Fork*. *Fork* beserta *join* (gabungan dari hasil *output fork*) dalam diagram berbentuk solid bar (batang penuh).

Tabel 2.2 Simbol-simbol *Activity Diagram*






| Nama | Simbol | Keterangan |
|----------------------------|---|--|
| <i>Activity</i> |  | Memperlihatkan bagaimana masing-masing kelas antar muka saling berinteraksi satu sama lain |
| <i>Action</i> |  | State dari sistem yang mencerminkan eksekusi dari suatu aksi |
| <i>Initial Node</i> |  | Menggambarkan awal dari suatu aktifitas yang berjalan pada sistem |
| <i>Activity Final Node</i> |  | Menggambarkan akhir dari suatu aktifitas yang berjalan pada sistem |
| <i>Fork Node</i> |  | Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran |
| <i>Join Node</i> |  | Menggambarkan hubungan antara dua state activity |

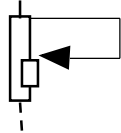
| | | |
|-----------------|---|--|
| <i>Decision</i> |  | Diagram yang mengindikasikan suatu kondisi dimana ada kemungkinan perbedaan transisi |
|-----------------|---|--|

2.4.3 Sequence Diagram

Diagram Class dan *Diagram Object* merupakan suatu gambaran model statis. Namun ada juga yang bersifat dinamis, seperti *Diagram Intraction*. *Diagram sequence* merupakan salah satu *Diagram Intraction* yang menjelaskan bagaimana suatu operasi itu dilakukan *message* (pesan). Diagram ini diatur berdasarkan waktu. Objek-objek yang berkaitan dengan proses berjalanya operasi diurutkan dari kiri kekanan berdasarkan waktu terjadinya dalam pesan yang terurut.

Tabel 2.3 Simbol-simbol Sequence Diagram

| Nama | Simbol | Keterangan |
|-----------------|---|---|
| <i>Actor</i> |  | Menggambarkan seseorang atau sesuatu yang berinteraksi dengan sistem |
| <i>Boundary</i> |  | Menggambarkan interaksi antara satu atau lebih actor dengan sistem |
| <i>Control</i> |  | Menangani tugas utama dan mengontrol alur kerja suatu sistem |
| <i>Entity</i> |  | Menggambarkan informasi yang disimpan oleh sistem (database) |
| <i>Message</i> |  | Komunikasi antar objek yang memuat informasi tentang aktifitas yang terjadi |

| | | |
|---------------------|---|---|
| <i>Self-message</i> |  | Menggambarkan pesan atau hubungan objek itu sendiri |
|---------------------|---|---|

2.4.4 Class Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (*atribut/property*) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (*metoda/fungsi*). *Class Diagram* menggambarkan struktur dan deskripsi *Class*, package dan objek beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain. *Class* memiliki tiga area pokok :

- Nama (dan *stereotype*)
- Atribut
- Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

- Private*, tidak dapat dipanggil dari luar class yang bersangkutan.
- Protected*, hanya dapat dipanggil oleh class yang bersangkutan dan anak-anak yang mewarisinya.
- Public*, dapat dipanggil oleh siapa saja.

Diagram Class mempunyai tiga macam relationship (hubungan), sebagai berikut :

- Association*

Suatu hubungan antara bagian dari dua kelas. Terjadi *Association* antara dua kelas jika salah satu bagian dari kelas mengetahui yang lainnya dalam melakukan suatu kegiatan. Di class diagram, sebuah *Association* adalah penghubung yang menghubungkan dua class.

- Agregation*

Suatu *Association* dimana salah satu kelasnya merupakan bagian dari suatu kumpulan. *Agregation* memiliki titik pusat yang mencakup keseluruhan bagian.

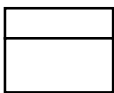


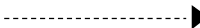
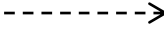
- Generalization*



Suatu hubungan turunan dengan mengasumsikan satu kelas merupakan suatu *superClass* (kelas super) dari kelas yang lain. *Generalization* memiliki tingkatan yang berpusat pada *superClass*.

Dari hubungan kelas dengan *Single Instance* (bagian) pada titik yang lain. *Multiplicity* berupa single number (angka tunggal) atau range number (angka batasan). Contoh, hanya bisa satu ‘*Customer*’ untuk setiap ‘*Order*’, tetapi satu ‘*Customer*’ hanya bisa memiliki beberapa ‘*Order*’. Tabel dibawah mengenai *Multiplicity* yang sering digunakan :

Setiap *diagram class* memiliki *class* (kelas), *association*, dan *multiplicity*. Sedangkan *navigability* (alur arah) dan *role* (kegiatan) merupakan optional (tidak diharuskan).

Tabel 2 4Simbol-simbol *Class Diagram*

| Nama | Simbol | Keterangan |
|---------------------------|---|---|
| <i>Class</i> |  | Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama |
| <i>Public Association</i> |  | Upaya untuk menghindari assosiasi dengan lebih dari 2 objek |
| <i>Generalization</i> |  | Hubungan dimana objek anak (<i>descendent</i>) berbagi perilaku dan struktur data dari objek yang ada diatasnya objek induk (<i>ancestor</i>) |
| <i>Realization</i> |  | Operasi yang benar-benar dilakukan oleh suatu objek |
| <i>Dependency</i> |  | Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri (<i>independent</i>) akan mempengaruhi elemen yang |

| | | |
|--------------------|---|--|
| | | bergantung padanya elemen yang tidak mandiri |
| <i>Association</i> |  | Menghubungkan antara objek satu dengan objek lainnya |
| <i>Agregation</i> |  | Suatu <i>association</i> dimana salah satu kelasnya merupakan bagian dari suatu kumpulan |

2.5 Basis Data

Secara umum definisi database adalah sekumpulan data yang berisi informasi mengenai satu atau beberapa object. Data dalam database tersebut biasanya disimpan dalam table yang saling berhubungan antara satu dengan yang lain. (Rina Musyawarah, 2007).

Basis data (*Database*) merupakan kumpulan data yang saling berhubungan satu dengan yang lainnya, tersimpan diperangkat keras computer dan digunakan perangkat lunak untuk memanipulasi data. Untuk mengelola basis data diperlukan perangkat lunak yang disebut Database Management system (DBMS). Adalah perangkat lunak system yang memungkinkan para pemakai membuat, memelihara, mengontrol dan mengakses basis data dengan praktis dan efisien. Database merupakan salah satu komponen yang penting dalam system informasi, karena merupakan basis dalam menyediakan informasi bagi pemakai. Komponen-komponen data yang dapat bertransformasi menjadi sebuah database diantaranya adalah :

- Haracters, merupakan bagian terkecil, dapat berupa karakter, numeric, huruf ataupun karakter khusus yang membentuk suatu item data.
- Field, suatu field menggambarkan suatu atribut dari record yang menunjukkan item dari data, seperti misalnya nama, alamat, dan lain sebagainya. Kumpulan suatu field membentuk suatu record.

- c. Record, kumpulan dari suatu field membentuk suatu record. Record menggambarkan suatu unit data individu tertentu. Kumpulan dari record 33 membentuk file, misalnya file personalia, tiap record dapat mewakili data tiap-tiap karyawan.
- d. File, file terdiri dari record-record yang menggambarkan dari satu-kesatuan data yang sejenis. Misalnya file mata kuliah berisi tentang data semua mata kuliah yang ada.
- e. Database, database merupakan sekumpulan file yang membentuk database.

2.5.1 Tujuan Basis Data

Basis data bertujuan untuk mengatur data sehingga diperoleh kemudahan, ketepatan dan kecepatan dalam pengambilan data kembali. Untuk mencapai tujuan, sarat sebuah data yang baik adalah :

- a. Redudansi dan inkositensi data, redudansi terjadi jika suatu informasi disimpan dibeberapa tempat.
- b. Kesulitan pengaksesan data, basis data memiliki fsilitas untuk melakukan pencarian informasi menggunakan query ataupun tools untuk melihat tabelnya.
- c. Multiple user, basis data memungkinkan pengguna data bersama-sama oleh banyak pengguna pada saat yang bersamaan atau pada saat yang berbeda.

2.6 Perancangan basis Data

2.6.1 Entity Relationship Diagram (ERD)

Diagram berhubungan entitas atau entity relation diagram merupakan model data berupa grafis, dalam pemodelan data konseptual yang menghubungkan antara penyimpanan. Model data sendiri merupakan sekumpulan cara, peralatan yang mendeskripsikan data-data yang hubunganya satu sama lain, serta batasan kosistensi. Model data terdiri dari model entitas dan hubungan relational diagram hubungan entitas digunakan untuk mengontruksikan model data konseptual, memodelkan struktur data dan hubungan data dan mengimplementasikan basis data secara logika maupun secara fisik DBMS (Database Management System). Dengan hubungan diagram entitas ini kita dapat menguji model dengan mengabaikan proses yang harus dilakukan. Diagram

hubungan entitas dapat membantu dalam menjawab persoalan tentang data yang diperlukan dan bagaimana data tersebut saling berhubungan.

2.6.2 Tranformasi ERD ke LRS

Tranformasi merupakan perubahan dari suatu bentuk ke bentuk yang lain. Dalam pembahasan kita berubah bentuk menjadi basis data fisik yang kita gunakan untuk membangun suatu sistem basis data, yakni ERD (*Entity Relationship Diagram*) ditranformasikan dalam bentuk basis data secara fisik.

Komponen ERD (*Entity Relationship Diagram*) ditranformasikan dalam bentuk tabel yang merupakan komponen utama pembentukan basis data. Atribut yang terdapat pada masing-masing entitas akan dinyatakan sebagai field atau kolom dari tabel yang sesuai.

2.6.3 Logical Relational Structure

Logical Relationship Structure dibentuk dengan nomor dari tipe record. Beberapa tipe record digambarkan oleh kotak empat persegi panjang dan nama yang unik. Beda LRS dengan diagram E-R nama tipe record berbeda diluar kotak field tipe record ditempatkan. *Logical Relationship Structure* terdiri dari *link-link* diantara tipe record. Link ini menunjukkan arah dari tipe satu record lainnya. Banyak link dari LRS yang diberi tanda field-field yang kelihatan pada kedua tipe link record. Dua metode yang dapat digunakan, dimulai dengan hubungan kedua model yang dapat dikonversikan ke LRS metode yang lain dimulai dengan ER-D dan langsung dikonversikan langsung ke LRS.

2.7 Pengujian Sistem

2.7.1 Filosofi Pengujian

Pengujian sistem merupakan proses mengeksekusi sistem perangkat lunak untuk menentukan apakah sistem perangkat lunak tersebut cocok dengan spesifikasi sistem dan berjalan sesuai dengan lingkungan yang diinginkan (Fatta, 2007). Pengujian sistem diasosiasikan dengan pencarian bug, ketidak sempurnaan program, kesalahan pada baris program yang menyebabkan kegagalan pada eksekusi sitem perangkat lunak (Fatta, 2007). Menemukan dan menghilangkan ketidak sempurnaan program disebut

debugging, yang berbeda dengan pengujian sistem yang berfokus pada pengidentifikasian adanya ketidak sempurnaan (Fatta, 2007).

2.7.2 Metode Pengujian Sistem

Beberapa tase-case harus dilakukan dengan beberapa perbedaan strategi transaksi, query, atau jalur navigasi yang mewakili penggunaan sistem yang tipikal, kritis atau abnormal. Isu kunci pada pengembangan sistem adalah pemilihan sekelompok tase-case yang cocok, sekecil, dan secepat mungkin, untuk meyakinkan perilaku sistem secara detail. Penguji harus mencakup unit testing, yang mengecek validasi dari prosedur dan fungsi-fungsi secara independen dari komponen sistem yang lain. Kemudian modul testing harus satu modul sudah berjalan dengan baik, termasuk eksekusi dari beberapa modul yang saling berelasi apakah sudah berjalan dengan karakteristik sistem yang diinginkan. Berikut ini kategori test yang dilakukan (Fatta, 2007).

b. Sub Testing

Adalah pengujian yang difokuskan pada pengujian struktur kendali sebelum semua modul dituliskan. Sistem perangkat lunak secara umum terdiri dari modul yang berelasi, baik secara hierarki maupun relasional. Pengujian ini penting untuk mengecek apakah struktur kendali sudah memetakan kinerja keseluruhan modul secara tepat. Jika ada modul yang berfungsi memanggil modul yang lain maka harus dipastikan semua modul bekerja sama dengan tepat.

c. Unit Testing

Pengujian unit digunakan untuk menguji setiap modul untuk menguji setiap modul yang menjamin setiap modul menjalankan fungsinya dengan baik. Ada 2 metode untuk melakukan unit testing yaitu :

1) Black Box Testing

Pada black box testing, cara pengujian hanya dilakukan dengan menjalankan atau mengeksekusi unit atau modul, kemudian diamati apakah hasil dari unit itu sesuai dengan proses bisnis yang diinginkan. Jika unit tidak sesuai outputnya maka untuk menyelesaikannya, diteruskan pada pengujian yang kedua, yaitu white box testing.

2) White Box Testing

White box testing adalah cara pengujian dengan melihat kedalam modul untuk meneliti kode-kode program yang ada, dan menganalisis apakah ada kesalahan atau tidak. Jika ada modul yang menghasilkan output yang tidak sesuai dengan proses bisnis yang dilakukan, maka baris-baris program, variable, dan parameter yang terlibat pada unit tersebut dicek satu persatu dan diperbaiki, kemudian di compile ulang.

d. Integration Testing

Setelah unit testing selesai selanjutnya adalah pengujian interaksi dari modul-modul yang menyusun system informasi untuk menjamin bahwa mereka bekerja dengan baik, integration testing terdiri dari :

1) Uji coba antar muka

Uji coba setiap fungsi dari antar muka

2) Uji coba scenario pengguna

Pastikan setiap scenario berjalan dengan baik

3) Uji coba aliran data

Uji coba proses dari langkah per langkah

4) Uji coba system antar muka

Pastikan data mengalir antar proses

e. Pengujian Sistem

Tes-tes sebelumnya berhubungan dengan bagaimana perangkat lunak system informasi berjalan dengan baik atau tidak. Komponen dari system informasi secara keseluruhan tidak hanya terdiri dari perangkat lunak saja, tetapi juga terdiri dari system tranmisi data, perangkat keras seperti computer, magnetic reader test untuk menjamin perangkat lunak bekerja dengan baik sebagai bagian dari keseluruhan system.

f. Acceptance Testing

Ketiga test yang dilakukan sebelumnya, merupakan test yang dilakukan oleh pengembang system. Karena pengguna akhir dari system memiliki pemahaman tentang system informasi dengan tingkat yang berbeda, maka seberapa jauh pengguna akhir dapat memahami dan menerima system harus diuji. Test inilah yang dinamakan user acceptance. Ada 2 tahapan dalam user acceptance yaitu :

1) Alpha testing

Alpha testing adalah test ini dilakukan untuk menjamin bahwa mereka menerima system, test dilakukan dengan menggunakan data test. Test ini sebenarnya simulasi dari pengguna system oleh pengguna akhir pada system sebenarnya tetapi dilakukan dengan data yang relative sedikit. Tujuannya adalah untuk melihat kemudahan pengguna perangkat lunak oleh pengguna akhir.

2) Beta testing

Beta testing adalah tahapan testing yang menentukan apakah system akan diterima atau harus dirancang ulang. Pada tahapan ini system diuji menggunakan data riil, bukan data test sehingga system dapat menggambarkan volume data yang sebenarnya yang harus diproses oleh system.

2.8 Perangkat Lunak Pendukung

2.8.1 Pemrograman Berbasis Web

Pada pemrograman web terdapat dua sisi programing. Yang pertama itu Client Side Programing, client side programing itu proses dilakukan pada sisi client (browser), script programnya dapat dilihat hanya dengan menggunakan view source.

Web adalah fasilitas hypertext yang mampu menampilkan data berupa text, gambar, suara, animasi, dan multimedia, dimana diantara data-data tersebut saling berkaitan dan berhubungan satu dengan yang lainnya. Untuk mempermudah dalam membaca data tersebut diperlukan sebuah web browser seperti explorer, mozilla dan lain-lainnya.

World wide web sering disebut www atau web adalah suatu metode untuk menampilkan informasi diinternet, baik berupa text, suara maupun video yang interaktif dan mempunyai kelebihan untuk menghubungkan (link) suatu dokumen dengan dokumen lainya (hypertext) yang dapat diakses melalui sebuah browser. Browser adalah perangkat lunak untuk mengakses halaman-halaman web seperti internet explorer, mozilla firefox, opera, safari dan lain-lain(Yuhafizar,2008).

2.8.2 HTML

Hyper Text Markup Language (HTML) sebenarnya bukan sebuah bahasa pemrograman, karena HTML adalah Bahasa markup. HTML digunakan untuk mark up

(penanda) terhadap suatu dokumen text.Symbol mark up yang digunakan oleh HTML ditandai dengan lebih kecil (<) dan tanda lebih besar (>).Kedua tanda ini disebut tag. Tag yang digunakan tanda penutup diberi karakter garis miring (</.>) (Binarso, Sarwoko, dan Bahtiar,2012).

HTML adalah kependekan dari Hyper Text Markup Language. Fasilitas hypertext merupakan metode yang menautkan (link) suatu dokumen ke dokumen lain melalui suatu text. HTML, merupakan halaman yang berbeda pada suatu situs internet atau web.Jadi, suatu situs terdiri atas beberapa halaman HTML atau webpage.Semakin menarik halaman webnya, semakin banyak website itu dikunjungi. Misalnya, dengan menampilkan gambar yang menarik, surat-surat, animasi huruf atau multimedia (Lia Kuswayatno,2006).

2.8.3 PHP

PHP adalah akronim dari hypertext preprocessor, yaitu suatu Bahasa pemrograman berbasis kode-kode (script) yang digunakan untuk mengolah suatu data untuk mengirimkannya kembali ke web browser menjadi kode HTML.(Diar Puji Oktavian,2010). Kode PHP memiliki ciri-ciri khusus, yaitu :

- a. Hanya dapat dijalankan menggunakan web server, misalnya Apache
- b. Kode PHP diletakkan dan dijalankan di web server
- c. Kode PHP dapat digunakan untuk mengakses database, seperti : MySQL, SQL, Oracle dan lain-lain.
- d. Merupakan software yang bersifat open source
- e. Gratis untuk di download
- f. Memiliki sifat multi platform, artinya dapat dijalankan menggunakan system operasi apapun, seperti Linux, Unix, Windows dan lain-lain.

2.8.4 Java Script

JavaScript adalah Bahasa pemrograman berbasis *prototype* yang berjalan di sisi *Client*.Jika kita berbicara dalam context web, sederhanya, kita dapat memahami *JavaScript* sebagai Bahasa pemrograman yang berjalan khusus untuk di browser atau halaman web agar halaman web menjadi lebih hidup. Kalau dilihat dari suku katanya

terdiri dari dua suku kata, yaitu *Java* dan *Script*. *Java* adalah Bahasa pemrograman berorientasi object, sedangkan *Script* adalah serangkaian intruksi program.

Secara fungsional, *JavaScript* digunakan untuk menyediakan akses *Script* pada object yang dibenamkan (*embedded*). Contoh sederhana dari penggunaan *JavaScript* adalah membuka halaman *pop up*, fungsi validasi pada form sebelum data dikirimkan ke *server*, merubah *image* kursor ketika melewati object tertentu, dan lain-lain.

JavaScript pertama kali dikenalkan oleh Netscape pada tahun 1995. Pada awalnya Bahasa yang sekarang disebut *JavaScript* ini dulunya dinamai “*LiveScript*” yang berfungsi sebagai bahasa sederhana untuk *browser* Netscape Navigator 2 yang sangat populer pada saat itu. Kemudian sejalan dengan sedang giatnya kerjasama antara Netscape dan Sun (pengembang Bahasa pemrograman “*Java*”) pada masa itu, maka Netscape memberikan nama “*JavaScript*” kepada bahasa tersebut pada tanggal 4 Desember 1995.

Pada saat yang bersamaan Microsoft sendiri mencoba untuk mengadaptasikan teknologi ini yang mereka sebut sebagai “*Jscript*” di *browser* milik mereka yaitu internet explorer 3. *JavaScript* sendiri merupakan modifikasi dari bahasa pemrograman C++ dengan pola penulisan yang lebih sederhana dari bahasa pemrograman C++.

2.8.5 Cascading Style Sheet

CSS (Cascading Style Sheet) adalah suatu bahasa yang di khususkan untuk mengatur gaya atau layout sebuah halaman web. CSS digunakan oleh pembuat halaman web dan juga pengakses halaman web, untuk mendefinisikan warna huruf layout dan aspek-aspek presentasi dokumen lainnya (Wahyu Sya’ban, 2010).

CSS memang didesain untuk memisahkan antara isi dokumen (yang ditulis menggunakan HTML atau bahasa lain yang sejenis) dengan bentuk presentasi dokumen (ditulis dalam CSS). Pemisahan ini diberikan keuntungan akan adanya peningkatan aksesibilitas isi, menyediakan fleksibilitas lebih, serta mereduksi kompleksitas berulang-ulangan pada struktur isi.

2.8.6 MySQL

MySQL adalah sebuah perangkat lunak system management basis data SQL atau yang dikenal dengan DBMS (Database Management System), database ini multithread, multi-user, MySQL AB membuat MySQL tersedia sebagai perangkat lunak gratis dibawah lisensi GNU General Public License (GPL), tetapi mereka juga menjual dibawah lisensi komersial untuk kasus-kasus yang bersifat khusus (Miftakhul Huda,2010).

MySQL (Muharam,2011) merupakan software database yang termasuk paling populer dilingkungan linux, kepopuleran ini karena ditunjang performasi query dari database nya yang saat itu bisa dikatakan paling cepat dan jarang bermasalah.

MySQL (My Structure Query Language) atau yang bias disebut “maissekuel” adalah sebuah program pembuat database yang bersifat open source, artinya siapa saja boleh menggunakannya dan tidk dicekal.

MySQL sebenarnya produk yang berjalan pada platform Linux, karena sifatnya yang open source, dia dapat dijalankan pada semua platform baik windows maupun linux.Selain itu, MySQL juga merupakan program pengakses database yang bersifat jaringan sehingga dapat digunakan untuk aplikasi multi-user (banyak pengguna).

MySQL adalah salah satu jenis database server yang sangat terkenal. Kepopuleranya disebabkan MySQL menggunakan SQL sebagai bahan dasar untuk mengakses database-nya. Selain itu, ia bersifat free pada berbagai platform (kecuali pada windows, yang bersifat shareware atau anda perlu membayar setelah melakukan evaluasi dan memutuskan untuk digunakan untuk keperluan produksi).

Sebagai program penghasil database, MySQL tidak dapat berjalan sendiri tanpa adanya sebuah aplikasi lain (interface). MySQL dapat di kung hamper semua program aplikasi baik yang open source seperti PHP maupun yang tidak, yang ada pada platform windows seperti visual basic, Delphi dan lainnya.

MySQL termasuk jenis RDBMS (Relational Database Management System).Itulah sebabnya istilah seperti table, baris, dan kolom diguanakan pada

MySQL. Pada MySQL, sebuah database mengandung satu atau sejumlah table. Table terdiri atas sejumlah baris dan setiap baris mengandung satu atau beberapa kolom.

2.9 Tools

Tools merupakan perangkat lunak yang dijadikan sebagai alat bantu. Adapun tools yang digunakan dalam pembuatan system pendaftaran siding proposal dan siding skripsi adalah sebagai berikut :

2.9.1 Php MyAdmin

PHP MyAdmin adalah suatu program open source yang berbasis web yang dibuat menggunakan PHP. Program ini digunakan untuk mengakses database MySQL, program ini mempermudah dan mempersingkat kerja kita. Dengan kelebihan, para pengguna awam tidak harus paham syntax-syntax SQL dalam membuat database dan table (Mahya,2008).

Data Definition Language (DDL) adalah suatu bahasa SQL yang berguna dalam pendefinisian data, yaitu pembuatan dan manipulasian table maupun database. Adapun beberapa bahasa SQL yang termasuk didalamnya adalah CREATE, ALTER, dan DROP atau (buat, ubah, dan hapus).

2.9.2 Xampp

XAMPP adalah sebuah software yang berfungsi untuk menjalankan website berbasis PHP dan menggunakan pengolahan data MySQL di computer local. XAMPP berperan sebagai server web pada computer anda. XAMPP juga dapat disebut sebuah Cpanel server virtual, yang dapat membantu anda melakukan priview sehingga dapat memodifikasi website tanpa harus online atau terakses dengan internet (Yogi Wicaksono,2008).