BAB II
LANDASAN TEORI

2.1. Pengembangan Sistem

Menurut Jogiyanto H.M. (2010:59), pengembangan sistem didefinisikan sebagai aktivitas untuk menghasilkan sistem informasi berbasis komputer untuk menyelesaikan persoalan (*problem*) organisasi atau memanfaatkan kesempatan (*opportunities*) yang timbul.

2.1.1. Tujuan Pengembangan Sistem

Adapun tujuan yang hendak dicapai dari tahap pengembangan sistem mempunyai maksud atau tujuan utama yaitu untuk memenuhi kebutuhan pemakaian sistem (*user*), untuk memberikan gambaran yang jelas dan menghasilkan pengembangan yang dapat memberikan kemudahan kepada pemograman komputer dan ahli-ahli teknik lainnya yang terlibat dalam pengguna sistem.

2.2. Sistem Informasi

Menurut Satzinger, Jackson, dan Burd (2010: 6), sistem informasi adalah sekumpulan dari komponen-komponen yang berkaitan yang mengumpulkan, memproses, dan menyediakannya sebagai *output* informasi yang dibutuhkan untuk menyelesaikan tugas-tugas bisnis.

2.3. Tugas Akhir

Penyusunan Tugas Akhir merupakan syarat wajib untuk menyelesaikan studi dalam program sarjana (strata 1). Tugas Akhir harus disusun oleh mahasiswa jika sudah memenuhi persyaratan yang sudah ditentukan oleh masing-masing program studi. Laporan Tugas Akhir dinilai layak uji jika memenuhi persyaratan yang berlaku.

Kekaryaan adalah implementasi dari eksplorasi konsep, ide, dan eksperimen yang bersifat akademis dan mandiri dari mahasiswa strata-1 dibantu dosen pembimbing sebagai fasilitator.Wujud Tugas Akhir kekaryaan berupa produk/jasa yang telah dikonsultasikan dan dilengkapi dengan deskripsi (Tim Penyusun Buku Pedoman Tugas Akhir Universitas Sahid Surakarta, 2011:3).

2.4. Rekayasa *Web*

Rekayasa *web* adalah sebuah aplikasi yang menggunakan pendekatan sistematis, disiplin, dan terukur untuk pengembangan, operasi dan pemeliharaan aplikasi berbasis *web* (*web-based application*). Sebagai catatan, kebanyakan literature tentang rekayasa *web* mengacu kepada situs *web*, aplikasi berbasis *web*, sistem berbasis *web*, dan apliaksi *web*.

Rekayasa *web* adalah subdisiplin dari rekayasa perangkat lunak yang membantu menyediakan metodologi untuk merancang, mengembangkan, memelihara, dan melibatkan aplikasi *web*. Beberapa metodologi telah diajukan dalam literature seperti OOHDM, OO-H, dan WebML. Beberapa metodologi lebih lanjut akan mendukung halaman-halaman *web* di dalam bentuk format lain, seperti WML untuk *mobile device* (Janner Simarmata,2010:1).

2.5. *CodeIgniter*

CodeIgniter merupakan *framework* PHP yang diklaim memiliki eksekusi tercepat dibandingkan dengan *framework* lainnya. *Codeigniter* bersifat *open source* dan menggunakan model kasus MVC (*Model View Controller*), yang merupakan model konsep modern *framework* yang digunakan saat ini (Agus Saputra, 2011).

Kelebihan framework codeigniterm, antara lain:

1) Mendukung PHP4 dan PHP5.
2) Berukuran kecil dan cepat.
3) Menggunakan konsep MVC.

2.6. PHP, HTML, *Web Server* dan MySQL

1) PHP

PHP adalah kependekan dari *PHP: HyperText Preprocessor* (suatu akronim rekursif) yang dibangun oleh Rasmus Lerdorf pada tahun 1994. PHP pada awal pengembanganya disebut sebagai kependekan dari *Personal Home Page*. PHP merupakan suatu bahas pemrograman sisi *server* yang dapat digunakan untuk membuat halaman Web dinamis (Antonius, 2010).

2) HTML

HTML (*HyperText Markup Language*) adalah bahasa standar untuk membuat halaman-halaman web, sedangkan PHP (*PHP Hypertext Preprocessor*)

berkedudukan sebagai tag dalam bahasa HTML. Model kerja HTML diawali dengan permintaan suatu halaman web oleh *browser*, dari *browser* permintaan dilanjutkan ke *webserver* yang kemudian mencarikan *file* yang diminta dan memberikan isinya ke *browser*. Perbedaannya jika menggunakan kode atau tag PHP adalah ketika berkas PHP yang diminta oleh *browser* didapatkan oleh *web server*, isinya segera dikirimkan ke mesin PHP dan mesin inilah yang memproses dan memberikan hasilnya (berupa kode HTML) yang kemudian akan dikirim ke *browser* oleh *webserver*. Secara khusus, PHP dirancang untuk membentuk aplikasi web dinamis (Kadir, 2008).

3) MySQL

MySQL adalah salah satu jenis *server* basis data yang sangat terkenal.Kepopulerannya disebabkan MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses basis datanya. Selain itu MySQL bersifat *open source* pada berbagai *platform*. MySQL termasuk jenis RDBMS (*Relational Database Management System*). Pada MySQL, sebuah basis data mengandung satu atau sejumlah tabel. Tabel sendiri terdiri atas sejumlah baris dan setiap baris mengandung satu atau beberapa kolom (Kadir, 2008).

4) *Web Server*

Menurut Yuniar Supardi (2010:181) "*Web server* merupakan perangkat lunak yang mengelola (mengatur) permintaan *user* dari *browser* dan hasilnya dikembalikan ke *browser*. Contoh *web server*, adalah IIS (*internet information services*) produk *microsoft corp*".

2.7. Analisis dan Perancangan Berbasis Objek

Menurut Nugroho (2009:4), UML (*Unified Modeling Language*) adalah Metodologi kolaborasi antara metoda-metoda Booch, OMT (*Object Modeling Technique*), serta OOSE (*Object Oriented Software Enggineering*) dan beberapa metoda lainnya, merupakan metodologi yang paling sering digunakan saat ini untuk analisa dan perancangan sistem dengan metodologi berorientasi objek mengadaptasi maraknya penggunaan bahasa "pemrograman berorientasi objek" (OOP).

2.7.1. *Use Case Diagram*

Rosa dan M. Shalahudin (2014:155), *use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih actor 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. Simbol-simbol *Use case Diagram* dapat dilihat pada Tabel 2.1.

Tabel 2.1 Simbol-Simbol *Use Case Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri.
2		*Generalization*	Hubungan generalisasi dan spesialisasi (umum-khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.
3		*Include*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan *use case* ini
4		*Extend*	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu.
5		*Association*	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan *actor*.
6		*Use Case*	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau actor.

2.7.2. *Class Diagram*

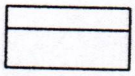
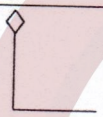
Rosa dan M. Shalahudin (2014:141), diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan method atau operasi. Berikut penjelasan atribut dan method :

1. Atribut merupakan variable-variabel yang dimiliki oleh suatu kelas.

2. Operasi atau method adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Simbol-simbol *Class Diagram* dapat dilihat pada Tabel 2.2.

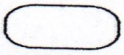
Tabel 2.2 Simbol-Simbol *Class Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Generalization*	Hubungan dimana objek anak *(descendent)* berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
2		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
3		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
4		*Aggregation*	Relasi antar kelas dengan makna semua-bagian (*whole-part*).
5		*Realization*	Operasi yang benar-benar dilakukan oleh suatu objek.
6		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri.
7		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

2.7.3. *Activity Diagram*

Rosa dan M. Shalahudin (2014:161), diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem. Simbol-simbol *Activity Diagram* dapat dilihat pada Tabel 2.3.

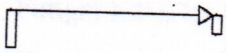
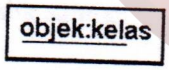
Tabel 2.3 Simbol-Simbol *Activity Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Initial Node*	Bagaimana objek dibentuk atau diawali.
2		*Action*	*State* dari sistem yang mencerminkan eksekusi dari suatu aksi.
3		*Actifity Final Node*	Bagaimana objek dibentuk dan dihancurkan.
4		*Actifity*	Memperlihatkan bagaimana masing-masing kelas antarmuka saling berinteraksi satu sama lain.
5		*Fork Node*	Satu aliran yang pada tahap tertentu berubah menjadi beberapa aliran.
6		*Decision*	Pilihan yang diambil untuk menunjukkan suatu keputusan yang lebih dari satu True / False.

2.7.4. *Sequence Diagram*

Rosa dan M. Shalahudin (2014:165), diagram sekuen menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dengan *massage* yang dikirimkan dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu. Simbol-simbol *Sequence Diagram* dapat dilihat pada Tabel 2.4

Tabel 2.4 Simbol-Simbol *Sequence Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*LifeLine*	Objek *entity*, antarmuka yang saling berinteraksi.
2		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
3		*Message*	Spesifikasi dari komunikasi antar objek yang memuat informasi-informasi tentang aktifitas yang terjadi
4		*Actor*	Orang, proses, atau sistem lain yang berinteraksi dengan system informasi dan mendapat manfaat dari system.
5		*Activation*	Menandakan ketika suatuobjek mengirim atau menerima pesan.
6		*Object*	Berpartipasi secara berurutan dengan mengirimkan atau menerima pesan.

2.7.5. *BlackBox*

Menurut Pressman (2010) *BlackBox* testing berfokus pada persyaratan fungsional perangkat lunak yang memungkinkan engineers untuk memperoleh set kondisi input yang sepenuhnya akan melaksanakan persyaratan fungsional untuk sebuah program.

BlackBox Testing bukan merupakan alternatif dari pengujian *WhiteBox* Testing. Sebaliknya, *BlackBox* Testing adalah pendekatan komplementer yang mungkin untuk mengungkap kelas yang berbeda dari kesalahan daripada metode *WhiteBox* Testing.

BlackBox Testing mencoba untuk menemukan kesalahan dalam kategori berikut :

a. Fungsi tidak benar atau hilang.
b. Kesalahan interface atau antarmuka.
c. Kesalahan dalam struktur data atau akses database eksternal.
d. Kesalahan kinerja atau perilaku.
e. Kesalahan inisialisasi dan terminasi.

Berikut ini contoh pengujian Kotak Hitam (*BlackBox* Testing) dapat dilihat pada Tabel 2.5.

Tabel 2.5 Pengujian *BlackBox*

No	Rancangan Proses	Hasil Yang Diharapkan	Hasil	Keterangan
1	Mengisi form login dan klik tombol login	Masuk halaman utama	Sesuai	Jika input benar
2	Klik menu edit profil sekolah	Membuka form input profil sekolah	Sesuai	
3	Mengisi form profil sekolah dan klik simpan	Data tersimpan dan membuka halaman utama	Sesuai	
4	Klik menu edit bidang kehalian	Membuka form input bidang keahlian	Sesuai	
5	Mengisi form input bidang keahlian dan klik simpan	Data tersimpan data muncul ditabel bidang keahlian	Sesuai	

2.7.6. Teknik Penentuan Skor

Untuk membantu dalam menganalisa data yang diperoleh dalam penelitian, maka penelitian ini menggunakan teknik penentuan skor. Teknik pengukuran skor yang akan digunakan adalah dengan modifikasi skala likert. Menurut Sugiyono (2009) Skala Likert digunakan untuk mengukur sikap, pendapat dan persepsi seseorang atau sekelompok orang tentang fenomena sosial. Penentuan presentase skor berdasarkan sekala likert mengunakan rumus sebagai berikut:

$$\% \text{ Skor} = \frac{\text{Skor Aktual}}{\text{Skor Ideal}}$$

Jawaban setiap pernyataan yang menggunakan skala likert mempunyai gradasi dari sangat positif sampai sangat negatif, kemudian untuk menentukan kriteria dalam membuat suatu kesimpulan dari hasil penelitian yang telah diperoleh, maka digunakan standarisasi nilai (presentase) yang dapat dilihat pada Tabel 2.6.

Tabel 2.6 Kriteria Skor Tanggapan Responden

No	Keterangan	Skor
1.	Sangat Setuju (SS)	5
2.	Setuju (ST)	4
3.	Ragu – Ragu (RG)	3
4.	Tidak Setuju (TS)	2
5.	Sangat Tidak Setuju (STS)	1

Sumber : Sugiyono (2009)