BAB III

LANDASAN TEORI

3.1 Aplikasi

Aplikasi adalah software yang dibuat oleh suatu perusahaan komputer untuk mengerjakan tugas-tugas tertentu, misalnya Microsoft Word, Microsoft Excel. (Dhanta (2009:32)).

Aplikasi adalah penerapan, penggunaan atau penambahan Dari pengertian diatas, dapat disimpulkan bahwa aplikasi merupakan software yang berfungsi untuk melakukan berbagai bentuk pekerjaan atau tugas-tugas tertentu seperti penerapan, penggunaan dan penambahan data. (Anisyah, 2000:30).

3.2 Web Application Development

Dari sistem yang akan dibangun, digunakan aplikasi berbasis web. Berikut adalah penjelasan dari tools yang digunakan untuk mengembangkan sebuah aplikasi berbasis web.

3.2.1 Personal Home Page

*Personal Home Page (*PHP) adalah skrip bersifat *server-side* yang ditambahkan ke dalam *HyperText Markup Language* (HTML). Skrip ini akan membuat suatu aplikasi dapat diintegrasikan ke dalam HTML sehingga suatu halaman *web* tidak lagi bersifat statis, namun menjadi bersifat dinamis. Sifat *server-side* berarti pengerjaan skrip akan dilakukan di server, baru kemudian hasilnya dikirim ke *browser* (Kurniawan, 2002).

Keunggulan dari sifatnya yang *server-side* tersebut antara lain :

1. Tidak diperlukan kompatibilitas *browser* atau harus menggunakan browser tertentu, karena server lah yang akan mengerjakan skrip PHP. Hasil yang dikirimkan kembali ke browser umumnya bersifat teks atau gambar saja sehingga pasti dikenal oleh *browser* apa pun.
2. Dapat memanfaatkan sumber-sumber aplikasi yang dimiliki oleh server, misalnya koneksi ke *database*.
3. Skrip tidak dapat "diintip" dengan menggunakan fasilitas *view* HTML *sourcode*.

Kelebihan PHP dapat melakukan semua aplikasi program *Common Gateway Interface* (CGI), seperti mengambil nilai form, menghasilkan halaman *web* yang dinamis, serta mengirim dan menerima *cookie*. PHP juga dapat berkomunikasi dengan layanan-layanan yang mengunakan protokol IMAP, SNMP, NNTP POP3, HTTP, dan lain-lain. Namun kelebihan yang paling signifikan adalah kemampuannya untuk dapat melakukan koneksi yang baik dengan berbagai macam *database*.

3.2.2 Web Server

Web server, untuk berkomunikasi dengan *client*-nya (*web browser*) mempunyai protokol sendiri, yaitu *hypertext tarnsfer protocol* (HTTP). Dengan protokol ini, komunikasi antar *web* server dengan *client*-nya dapat saling dimengerti dan lebih mudah (Azmi, 2012).

Web server bisa di deskripsikan dengan formula"*Web Server = Platform + Software + Information*". *Web server* haruslah komputer dengan

koneksi ke *internet*, dengan sistem *software* untuk menjalankan komputer dan untuk dapat terhubung dengan sistem lain di *internet* (Yeager & McGrath, 1996).

3.3 Disposisi

Disposisi dari sudut pandang administrasi berarti pendapat seorang pejabat mengenai urusan yang termuat dalam suatu surat dinas, yang langsung dituliskan pada surat yang bersangkutan atau pada lembar khusus. Melihat defenisi tersebut jelas sekali bahwa disposisi haruslah memberikan informasi yang jelas berupa instruksi, petunjuk dan arahan dari pejabat lebih tinggi apa yang harus dilakukan kepada bawahannya. (Sumarmo, 2010.)

Lembar disposisi secara umum berisi informasi nonverbal untuk diterjemahkan secara seksama dan ditindaklanjuti. Disposisi merupakan petunjuk singkat tentang tindak lanjut(penyelesaian) terhadap suatu urusan atau surat masuk. Disposisi dibuat oleh pimpinan untuk staf atau bawahan sesuai dengan bidang keahlian atau kewenangannya. Tujuan pembuatan disposisi ialah agar staf dapat menindaklanjuti atau menyelesaikan suatu urusan atau surat masuk sesuai dengan yang dikehendaki oleh pimpinan. Tindak lanjut dapat berupa surat balasan, tindakan-tindakan lain dalam rangka menyelesaikan urusan tersebut.

Sebelum diserahkan kepada pimpinan, surat masuk terlebih dahulu dikendalikan oleh bagian administrasi dan diberi lembar disposisi. Pengisian lembar disposisi yang menyangkut masalah pengagendaan seperti indeks, kode, nomor urut, dan data-data tentang surat dilakukan oleh bagian administrasi. Indeks, kode, dan nomor urut berisi kata/sandi yang bisa digunakan untuk merunut kembali dimana letak surat tersebut dalam agenda atau pengarsipan. Selanjutnya pimpinan

mendisposisikan siapa yang diberi tugas atau tanggung jawab untuk menindaklanjuti, serta bagaimana intruksi-intruksinya.

3.4 Notifikasi

Notifikasi adalah pemberitahuan yang tertuju untuk kita dari seseorang, tujuan notifikasi untuk mengingatkan kita pada hal tersebut. Dengan adanya notifikasi, orang mengerti bahwa ada sesuatu yang dikirimkan kepada kita. Notifikasi biasanya diletakan ditempat yang sering terlihat supaya seseorang mengerti bahwa ada pemberitahuan baru. Notifikasi akan hilang apabila pesan tersebut telah terbaca. (Firdaus, 2007)

3.5 Analisa dan Desain Sistem

Menurut (Ladjamudin, 2005), Analisa sistem yang ada sangat bergantung pada teori umum sebagai sebuah landasan konseptual. Bertujuan untuk memperbaiki fungsi di dalam sistem yang sedang berjalan agar menjadi lebih efisien, mengubah sasaran sistem yang sedang berjalan.

3.6 Object Oriented Design (OOD)

OOD adalah metode desain meliputi proses dekomposisi berorientasi objek dan notasi untuk menggambarkan sebuah model baik secara fisik dan logis serta statis dan dinamis dari sebuah sistem pada desain (Booch, 1994).

3.7 Unified Modeling Language (UML)

UML adalah bahasa standar yang digunakan untuk menjelaskan dan memvisualisasikan artifak dari proses analisis dan disain berorientasi objek. UML

menyediakan standar pada notasi dan diagram yang bisa digunakan untuk memodelkan suatu sistem. Ada beberapa diagram yang disediakan dalam UML (Sholiq, 2006), antara lain:

a. Diagram *use case (use case diagram)*

b. Diagram aktivitas *(activity diagram)*

c. Diagram sekuensial *(sequence diagram)*

d. Diagram kolaborasi *(collaboration diagram)*

e. Diagram kelas *(class diagram)*

f. Diagram *statechart (statechart diagram)*

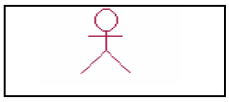
g. Diagram komponen *(component diagram)*

h. Diagram *deployment (deployement diagram)*

Terdapat notasi pada UML, antara lain :

A. Actor

Actor adalah segala sesuatu yang berinteraksi dengan sistem aplikasi komputer. Jadi *actor* ini bisa berupa orang, perangkat keras, atau mungkin juga objek lain dalam sistem yang sama. Biasanya yang dilakukan oleh *actor* adalah memberikan informasi pada sistem dan memerintahkan sistem untuk melakukan sesuatu. Pada Gambar 3.1 akan ditunjukan notasi *actor*.

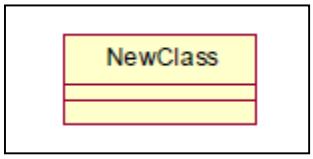


Gambar 3. 1 Notasi *Actor*

B. Class

Class merupakan pembentuk utama dari sistem berorientasi objek karena *class* menunjukan kumpulan objek yang memiliki atribut dan operasi yang sama.

Class digunakan untuk mengimplementasikan *interface*. Pada Gambar 3.2 akan ditunjukan notasi *class.*



Gambar 3. 2 Notasi *Class*

C. Interface

Interface merupakan kumpulan operasi tanpa implementasi dari suatu *class.* Implementasi operasi dalam *interface* dijabarkan oleh operasi dalam *class.* Oleh karena itu keberadaan *interface* selalu disertai oleh *class* yang mengimplementasikan operasinya. *Interface* ini merupakan salah satu cara mewujudkan prinsip enkapsulasi dalam objek. Pada Gambar 3.3 akan ditunjukan notasi *interface*.

Gambar 3. 3 Notasi *Interface*

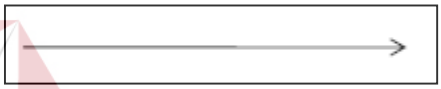
D. Use case

Use case menjelaskan urutan kegiatan yang dilakukan *actor* dan sitem untuk mencapai suatau tujuan tertentu. Walaupun menjelaskan kegiatan namun *use case* hanya menjelaskan apa yang dilakukan oleh *actor* dan sistem, bukan bagaimana *actor* dan sistem melakukan kegiatan tersebut. Pada Gambar 3.4 akan ditunjukan notasi *use case*.

Gambar 3. 4 Notasi *Use Case*

E. Interaction

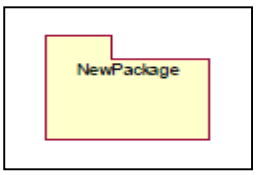
Interaction digunakan unutk menunjukan baik aliran pesan atau informasi antar objek. Biasanya *interaction* ini dilengkapi juga dengan teks bernama *operation signature* yang tersusun dari nama operasi, parameter yang dikirim dan tipe parameter yang dikembalikan. Pada Gambar 3.5 akan ditunjukan notasi *interaction.*



Gambar 3. 5 Notasi *Interaction*

F. Package

Package adalah *container* atau wadah konseptual yang digunakan untuk mengelompokkan elemen-elemen dari sistem yang sedang dibangun, sehingga bisa dibuat model yang lebih sederhana. Tujuannya adalah untuk mempermudah penglihatan *(visibility)* dari model yang sedang dibangun.

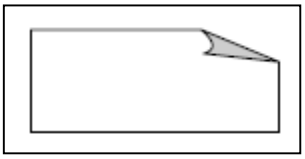


Gambar 3. 6 Notasi *Package*

G. Note

Note digunakan untuk memberikan keterangan dan komentar tambahan dari suatu elemen sehigga bisa langsung terlampir dalam model. *Note* ini bisa

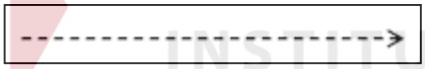
ditempelkan ke semua elemen notasi yang lain. Pada Gambar 3.7 akan ditunjukan notasi *note*.



Gambar 3. 7 Notasi *Note*

H. Dependency

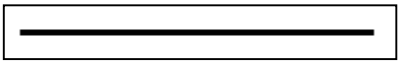
Dependency merupakan relasi yang menunjukkan bahwa perubahan pada salah satu elemen memberi pengaruh pada elemen lain. Elemen yang ada di bagian tanda panah adalah elemen yang tergantung pada elemen yang ada di bagian tanpa tanda panah. Pada Gambar 3.10 akan ditunjukan notasi *dependency*.



Gambar 3. 8 Notasi *Dependency*

I. Association

Association menggambarkan navigasi antar *class (Navigation)*, beberapa banyak objek lain yang bisa berhubungan dengan satu objek *(Multiplicity* antar *class)*, dan apakah suatau *class* menjadi bagian dari *class* lainnya *(Aggregation)*.

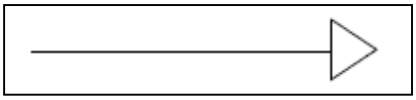


Gambar 3. 9 Notasi *Association*

J. Generalization

Generalization menunjukkan hubungan antara elemen yang lebih umum ke elemen yang lebih spesifik. Dengan *generalization*, *class* yang lebih spesifik *(subclass)* akan menurunkan atribut dan operasi dari *class* yang lebih umum

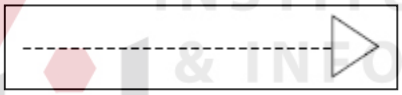
(superclass), atau *"subclass is a superclass"*. Dengan menggunakan notasi *generalization* ini konsep *inheritance* dari prinsip hirarki dimodelkan. Pada Gambar 3.10 akan ditunjukan notasi *generalization*.



Gambar 3. 10 Notasi *Generalization*

K. Realization

Realization menunjukkan hubungan bahwa elemen yang ada di bagian tanpa panah akan merealisasikan apa yang dinyatakan oleh elemen yang ada di bagian dengan panah. Misalnya merealisasikan *package*, *component* merealisasikan *class* atau *interface*. Pada Gambar 3.11 akan ditunjukan Notasi *realization.*



Gambar 3. 11 Notasi *Realization*

3.8 Sistem Basis Data

Menurut (Kusrini, 2007), Basis data adalah kumpulan data yang memiliki relasi antar entitas, sehingga adanya basis data ini mempunya tujuan untuk mempermudah perolehan data dalam waktu yang singkat dan ketepatan data yang diperoleh.

Kelebihan sistem basis data :

1. Pengendalian terhadap redudansi data.
2. Mencegah ketidakkonsistenan data.
3. Keamanan data dapat terjaga, yaitu data dapat dilindungi dari pemakai yang tidak berwenang.

4. Integritas data dapat dipertahankan.
5. Data dapat dipergunakan bersama-sama.
6. Menyediakan *recovery*.
7. Memudahkan penerapan standarisasi.
8. Data bersifat mandiri (data *independence*).
9. Keterpaduan data terjaga, memelihara keterpaduan data berarti data harus akurat. Hal ini sangat erat hubungannya dengan pengontrolan kerangkapan data dan pemeliharaan keselarasan data.

Kekurangan sistem basis data :

1. Diperlukan tempat penyimpanan yang besar.
2. Diperlukan tenaga yang terampil dalam mengolah data.
3. Kerusakan sistem basis data dapat mempengaruhi departemen yang terkait.
4. Kompleksitas yang tinggi
5. Ongkos konversi dari sistem yang lama ke sistem baru

3.9 Database Management System

Menurut (Bambang Haryanto, 2004), Sistem manajemen basis data adalah perangkat lunak untuk mendefinisikan, menciptakan, mengelola, dan mengendalikan pengaksesan basis data.

DBMS bertujuan menyediakan lingkungan yang nyaman dan efisien untuk penyimpanan data dan pengambilan data dari basis data. DMBS sangat berperan memberi abstraksi data tingkat tinggi ke pemakai.

3.9.1 Bahasa-Bahasa Yang Terdapat Dalam DBMS

1. *Data Definition Language* (DDL)

Menurut (Kristanto, 1994), Pola skema basis data di spesifikasikan dengan satu set definisi yang di ekspresikan dengan satu bahasa khusus yang disebut DDL. Hasil kompilasi perintah DDL adalah satu set tabel yang disimpan di dalam *file* khusus yang disebut data *dictionary/directory.*

2. *Data Manipulation Language* (DML)

   Bahasa yang memperbolehkan pemakai mengakses atau memanipulasi data sebagai yang diorganisasikan sebelumnya model data yang tepat.

3. *Query*

   Pernyataan yang diajukan untuk mengambil informasi. Merupakan bagian DML yang digunakan untuk pengambilan informasi.

3.9.2 Fungsi DBMS

1. *Data Definition*

   DBMS harus dapat mengolah data *definition* atau pendefinisian data.

2. *Data Manipulation*

   DBMS harus dapat menangani permintaan-permintaan dari pemakai untuk mengakses data.

3. *Data Security dan Integrity*

   DBMS dapat memeriksa *security* dan *integrity* data yang didefinisikan oleh DBA.

4. *Data Recovery* dan *Concurrency*

   a. DBMS harus dapat menangani kegagalan-kegagalan pengaksesan basis data yang dapat disebabkan oleh kesalahan sistem, kerusakan *disk*, dan sebagainya.

   b. DBMS harus dapat mengontrol pengaksesan data yang konkuren yaitu bila satu data diakses secara bersama-sama oleh lebih dari satu pemakai pada saat yang bersamaan.

5. *Data Dictionary*

   DBMS harus menyediakan *data dictionary* atau kamus data.