BAB II

LANDASAN TEORI

2.1. Tinjauan Pustaka

Membangun sebuah *website* yang baik agar sesuai dengan kebutuhan yang diinginkan penulis dan Program Studi Ilmu Komunikasi Universitas Sahid Surakarta, maka diperlukan referensi *website* yang sama-sama menyajikan informasi seputar Ilmu Komunikasi. Kemungkinan referensi ini juga di ambil dari beberapa situs *website* perguruan tinggi dan lembaga Ilmu Komunikasi lainnya yang selama ini menyajikan informasi tentang Ilmu Komunikasi. Seperti *website* Ilmu Komunikasi UMS, *website* Ilmu Komunikasi UNS, dan *website* Ilmu Komunikasi UNISRI.

Website Ilmu Komunikasi Universitas Muhammadyah Surakarta (UMS) yang dapat diakses melalui www.komunikasi.ums.ac.id, *website* ini dari penampilan terlihat *simple* tidak terlalu banyak gambar. Fitur menunya juga tidak terlalu banyak dan menarik untuk dilihat. Contoh tampilan website Ilmu Komunikasi UMS pada Gambar 2.1.

Gambar 2.1 *Website* Ilmu Komunikasi UMS

Website Ilmu Komunikasi Universitas Sebelas Maret (UNS) yang dapat diakses melalui www.komunikasi.fisip.uns.ac.id, website ini bagus dan mempunyai fasilitas dihalaman *website* seperti fasilitas yang ada di Ilmu Komunikasi UNS, informasi seputar Program Studi Ilmu Komunikasi yang lengkap dan *website* Ilmu Komunikasi UNS juga dilengkapi dengan fasilitas *download* sehingga mempermudah mahasiswa untuk memperoleh data yang ingin mereka dapat. Contoh tampilan *website* Ilmu Komunikasi UNS pada Gambar 2.2.

Gambar 2.2 *Website* Ilmu Komunikasi UNS

Website Ilmu Komunikasi Universitas Slamet Riyadi (UNISRI) yang dapat diakses melalui www.fisip.unisri.ac.id, *website* ini mempunyai banyak fitur menu sehingga tampilan website terlihat penuh dan terlalu banyak *content-content* seperti gambar. Tetapi jika dilihat secara keseluruhan warna tampilan *website* tidak terlalu mencolok dan menarik untuk dilihat. Contoh tampilan website Ilmu Komunikasi UNISRI pada Gambar 2.3.

Gambar 2.3*Website* Ilmu Komunikasi UNISRI

Website Ilmu Komunikasi USS (Universitas Sahid Surakarta), sebuah situs *website* ilmu komunikasi Universitas Sahid Surakarta yang mempunyai beberapa *menu* seperti: beranda, agenda, akademik mempunyai *submenu* (kurikulum, mata kuliah, kalender akademik) alumni, fasilitas mempunyai *submenu* (laboratorium, perpustakaan, radio, uk tv), Profil, visi dan misi, struktur organisasi, staff pengajar, kontak kami.

Pembuatan *website* ini, diperlukan beberapa teori dan definisi yang berhubungan dengan penyelesaian Tugas Akhir. Beberapa hal di antaranya: *Internet*, *Web Server*, *Website*, HTML, PHP, CSS, *Web Browser*, *Adobe Dreamweaver*, Basis Data, MySQL, Analisis dan Perancangan Sistem, UML ***(****Unified Modeling Language****).***

2.2. *Internet*

2.2.1. Sejarah Perkembangan *Internet*

Secara etimologis, *internet* berasal dari bahasa Inggris, yaitu *internet* berarti berarti jaringan sehingga dapat diartikan sebagai hubungan antarjaringan. *Internet* merupakan suatu media informasi yang berjalan dalam suatu komputer. Akan tetapi, tidak semua komputer yang ada bisa berhubungan karenasuatu komputer dapat

dikatakan sebagai *internet* jika sudah terhubung dengansuatu jaringan. Sejarah terciptanya *internet* dimulai di Amerika, saat itu dalamkeadaan perang.Sebelum *internet* ada, ARPAnet (US *Defense Advanced Research Project Agency*) atau Departemen Pertahanan Amerika pada tahun 1969 membuatjaringan komputer yang tersebar untuk menghindari terjadinya infor masiterpusat, yang apabila terjadi perang mudah dihancurkan. Bila satu bagian darisambungan *network* terganggu serangan musuh, jalur yang melalui sambunganitu secara otomatis dipindahkan ke sambungan lainnya. Setelah itu *internet* digunakan oleh kalangan akademis (UCLA) untuk keperluan penelitian danperkembangan teknologi. Selanjutnya pemerintah Amerika Serikat memberikan ke arah komersial pada awal tahun 1990 (M.Suyanto, 2005).

2.2.2. Pengertian *Internet*

Internet adalah sebuah jaringan global dari jaringan komputer yang menghubungkan sumberdaya – sumberdaya bisnis, pemerintahan, dan institusi pendidikan menggunakan protokol TCP/IP (*Transmission Control Protocol*/ *Internet Protocol*). Dari sekitar 50 juta pemakai di tahun 1997 akan meningkat menjadi 750 juta pemakai pada tahun 2007. Peralatan tanpa kabel yang mengakses *internet* dan integrasi televisi dan komputer akan menjadi *internet* dapat mencapai setiap rumah, lembaga bisnis, sekolah, pemerintah, dan organisasi lainya. Aplikasi-aplikasi multimedia *internet* dikategorikan menjadi 3 kategori, yaitu pencarian, komunikasi dan kolaborasi.Pencarian meliputi "*browsing*" dan pengambilan informasi. Pemakai mempunyai kemampuan untuk melihat dokumen dan *mendownload* apa saja yang mereka butuhkan. *Internet* juga menyediakan saluran komunikasi relatif murah dan cepat yang menjangkau pesan yang ditayangkan pada papan buletin, sampai pertukaran informasi kompleks diantara organisasi satu dengan organisasi lainnya.Aplikasi kolaborasi dapat digunakan dalam konferensi jarak jauh dan layar bersama bersama pada sistem yang mendukung grup.Beberapa *tool* kolaborasi yang disebut "*groupware*" dapat digunakan pada *internet* dan jaringan lainnya (M.Suyanto, 2005).

2.3. *Website*

Website (situs *web*) adalah merupakan alamat (URL) yang berfungsi sebagai tempat penyimpanan data dan informasi dengan berdasarkan topik tertentu. *Web* menyediakan jaringan dimana-mana dan murah.*Website* mempunyai pengguna berdasarkan *software web browser* yang distandarisasi yang berjalan pada berbagai komputer biasa (Jhanner Shimarta, 2006).

2.4. PHP (*Personal Home Page)*

PHP merupakan singkatan dari PHP *Hypertext Preprocessor*. Ia merupakan bahasa berbentuk skrip yang ditempatkan dalam *server* dan diproses di *server*. Hasilnya yang dikirimkan ke klien, tempat pemakai menggunakan *browser*.

Secara khusus, PHP dirancang untuk membentuk aplikasi *web* dinamis. Artinya, ia dapat membentuk suatu tampilan berdasarkan permintaan terkini. Misalnya, Anda bisa menampilkan isi *database* ke halaman *web*. Pada prinsipnya PHP mempunyai fungsi yang sama dengan skrip-skrip seperti ASP (*Active Server Page*), *Cold Fusion*, ataupun *Perl*. Namun, perlu diketahui bahwa PHP sebenernya bisa dipakai secara *command line*. Artinya, skrip PHP dapat dijalankan tanpa melibatkan *webserver* maupun *browser* (Abdul Kadir, 2008).

2.5. MySQL

Menurut Heni A. Puspitosari (2011) jika ingin membuat *website* yang interaktif dan dinamis, perlu adanya media penyimpanan data yang flieksibel dan mudah untuk diakses.Dalam bahasa pemrograman sering ada istilah *database*. *Database* adalah kumpulan data-data yang saling terkait, tersimpan, dan mudah untuk diakses.

Salah satu program yang dapat digunakan sebagai *database* adalah MySQL. MySQL merupakan salah satu *software* untuk *databaseserver* yang banyak digunakan, MySQL bersifat *opensouce* dan menggunakan SQL. MySQL bisa dijalankan di berbagai *platform* misalnya *Windows*, *Linux*, dan lain sebagainya.

2.6. *AdobeDreamweavear*

Adobe Dreamweaver merupakan salah satu program aplikasi yang digunakan untuk membangun sebuah *website*, baik secara grafis maupun dengan menulis kode sumber secara langsung.

Adobe Dreamweaver CS4 memudahkan pengembang *website* untuk mengelola halaman-halaman *website* dan aset-asetnya, baik gambar (*image*), animasi *flash*, *video*, suara dan lain sebagainya. Selain itu, *Adobe Dreamweaver* CS4 juga menyediakan fasilitas untuk melakukan pemrogaman *scripting*, baik ASP (*Active Server Page*), JSP (*Java Server Page*), PHP (PHP *Hypertext Preprocessor*), *Java Script* (js), *Cold Fusion*, CSS (*Cascading Style Sheet*), XML (*Extensible Markup Language*) dan lainnya (Wahana Komputer, 2010).

2.7. *Web Server*

Web Server dibutuhkan untuk melakukan pemrograman yang membutuhkan teknologi *server-side*, seperti PHP, *Cold Fusion*, ASP dan JSP. Oleh karena itu, *web server* merupakan perangkat yang penting untuk melakukan uji coba pemrograman *server-side* yang anda buat. Dengan demikian, anda tidak perlu melakukan *upload* data ke sebuah *hosting* hanya untuk mengetahui hasil pemrograman untuk sisi *server*. Pemrograman *server-side* biasanya digunakan untuk mengakses *database* yang terdapat pada *server hosting*, sehingga pengembang dengan mudah mengelola *content website* dengan mudah tanpa mengubah halaman-halaman *web* (Wahana Komputer, 2010).

2.8. HTML (*Hypertext Markup Language*)

HTML adalah kependekan dari *Hyper Text Markup Language*, merupakan sebuah bahasa *scripting* yang berguna untuk menuliskan halaman *web*.Pada halaman *web*, html dijadikan sebagai bahasa *script* dasar yang berjalan bersama berbagai bahasa *scripting* pemrograman lainya.

Semua tag-tag HTML bersifat dinamis, artinya kode html tidak dapat dijadikan sebagai *file executable* program. Hal ini disebabkan html hanyalah sebuah bahasa *scripting* yang dapat berjalan apabila dijalankan di dalam *browser*

(pengaksesan *web*), *browser-browser* yang mendukung html antara lain adalah *Internet Explorer*, *Netscape Navigator*, *Opera*, *Mozilla* dan lain-lain. Jadi pada saat ingin membuka halaman yang berasal dari html anda dapat melihat bentuk pengkodeanya dengan cara mengklik menu *View source*, maka disana akan ditampilkan semua tag beserta isi dari halaman *web* tersebut (Bunafit Nugroho, 2004).

2.9. CSS (*Cascading Style Sheets*)

CSS (*Cascade Style Sheet*) adalah sebuah fitur yang diperkenalkan sejak HTML versi 4.0 dan berfungsi untuk menangani masalah tampilan pada HTML seperti jenis, ukuran dan warna *font*, posisi teks, batas tulisan atau *margin*, warna *background*, dan sebagainya.

Dari sisi manajemen dan perawatan, penggunaan CSS dipandang lebih praktis karena para *web developer* tidak perlu membuka setiap *file* dalam sebuah situs untuk melakukan perubahan.Sebagai contoh, katakanlah dalam sebuah situs digunakan *font* berjenis *arial*.

Hal penting yang perlu diperhatikan adalah cara meletakkan CSS dan juga bahasa berbasis *web* lain untuk memudahkan menajemen *file*, *editing*, dan *maintenance*. Banyak di antara *programmer web* yang belum menyadari aspek-aspek penting ini menyisipkan CSS, *JavaScript*, *VBScript*, PHP, maupun ASP langsung ke dalam dokumen HTML (*embedded script*) (Madcoms, 2008).

2.10. Basis Data (*Database*)

Menurut Fathansyah (2007) Basis Data (*Database*) terdiri atas 2 kata, yaitu Basis dan Data. Basis kurang lebih dapat diartikan sebagai markas atau gudang, tempat bersarang/berkumpul. Sedangkan Data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keaadaan, dan sebagainya, yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya.

Basis Data sendiri dapat didefinisikan dalam sejumlah sudut pandang seperti:

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan yang disimpan secara bersama sedemikian rupa dan tanpa pengulangan (redudansi) yang tidak perlu, untuk memenuhi berbagai kebutuhan.
3. Kumpulan *file*/tabel/arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronis.

2.11. Analisis dan Perancangan Sistem Berbasis Objek

Pemrograman berorientasi objek adalah suatu cara baru dalam berpikir serta berlogika dalam menghadapi masalah-masalah yang akan dicoba diatasi dengan bantuan komputer. Tidak seperti pendahulunya (pemrograman terstruktur), pemrograman berorientasi objek mencoba melihat permasalahan lewat pengamatan dunia nyata dimana setiap objek adalah entitas tunggal yang memiliki kombinasi struktur data dan fungsi tertentu Nugroho (2002:1). Orietasi objek adalah sebuah pendekatan untuk mengembangkan software sistem yang bedasarkan pada item data dan atribut dan operasi operasi yang menjelaskanya Britton dan Doake (2001).

Analisis dan perancangan metodologi berorientasi objek menggunakan *diagram* UML. UML adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek Fowler (2005).

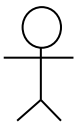
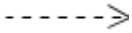
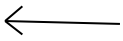
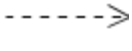
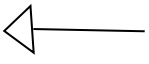
Kelebihan dari perancangan berorientasi objek menurut Britton dan Doake (2001) adalah sebagai berikut :

a. *Maintainable* : pemeliharan
b. *Testable* : dicoba
c. *Reueseable* : dapat digunakan kembali
d. *Albe to cope with large and complex system* : dapat berkerjasama dengan system yang luas dan kompleks.

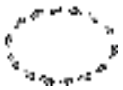
2.11.1. *Use Case Diagram*

Use case diagram adalah teknik untuk merekam persyaratan fungsional sebuah sistem. *Use case* mendeskripsikan interaksi tipikal antara para pengguna sistem dengan sistem itu sendiri, dengan memberi sebuah narasi tentang bagaimana sistem tersebut digunakan Fowler (2005:141). *Diagram Use Case* dekat kaitannya dengan kejadian-kejadian. Kejadian (scenario) merupakan contoh apa yang terjadi ketika seseorang berinteraksi dengan sistem.*Use Case Diagram* dibuat untuk memvisualisasikan atau menggambarkan hubungan antara *Actor* dan *UseCase*. Simbol-simbol yang digunakan pada *use case* diagram ditunjukan pada Tabel 2.1.

Tabel 2.1. Simbol-simbol *Use Case Diagram*

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.

Lanjutan Tabel 2.1. Simbol-simbol *Use Case* Diagram

NO	GAMBAR	NAMA	KETERANGAN
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.
9		*Collaboration*	Interaksi aturan-aturan dan elemen lain yang bekerja sama untuk menyediakan prilaku yang lebih besar dari jumlah dan elemen-elemennya (sinergi).

2.11.2. *Class Diagram*

Class Diagram mendeskripsikan jenis-jenis objek dalam sistem dan berbagai macam hubungan statis yang terdapat diantara mereka.*Class Diagram* juga menunjukkan properti dan operasi sebuah class dan batasan-batasan yang terdapat dalam hubungan-hubungan objek tersebut Martin Fowler (2005:53).

Diagram kelas mempunyai 3 macam *relationalships* (hubungan), sebagai berikut:

a. Association

Suatu hubungan antara bagian dari dua kelas, terjadiassociationantara dua kelas jikasalah satu bagian dari kelas mengetahui yang lainnya dalam melakukan suatu kegiatan. Di dalam *diagram*, sebuah association adalah penghubung yang menghubungkan dua kelas.

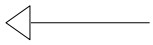
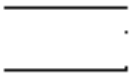
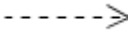
b. Aggregation

Suatu *association* dimana salah satu kelasnya merupakan bagian dari suatu kumpulan, *aggregation* memiliki titik pusat yang mencakup keseluruhan bagian, sebagai contoh: *Order Detail* merupakan kumpulan dari *Order.*

c. Generalization

Suatuhubungan turunan dengan mengasumsikan satu kelas merupakan suatu *super class* (kelas super) dari kelas yang lain. *Generalization* memiliki tingkatan yang berpusat pada*super class,* contoh: *Payment* adalah*superclass* dari *Cash, Check, Credit*. Simbol-simbol yang digunakan pada *use case* diagram ditunjukan 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		*Nary Association*	Upaya untuk menghindari asosiasi dengan lebih dari 2 objek.
3		*Class*	Himpunan dari objek-objek yang berbagi atribut serta operasi yang sama.
4		*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor.
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.

2.11.3. *Activity Diagram*

Activity Diagram adalah teknik untuk menggambarkan logika prosedural, proses bisnis, dan jalur kerja, dalam beberapa hal, *diagram* ini memainkan peran mirip sebuah *diagram* alir, tetapi perbedaan prinsip antara *diagram* ini dan notasi

diagram alir adalah *diagram* ini mendukung *behavior parallel* Martin Fowler, (2005:163). Simbol-simbol yang digunakan pada *use case* diagram ditunjukan pada Tabel 2.3.

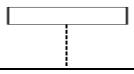
Tabel 2.3 Simbol-Simbol *Activity Diagram*

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

2.11.4. *Sequence Diagram*

Sequence diagram adalah penjabaran *behavior* sebuah skenario tunggal. *Sequence diagram* menunjukkan sejumlah objek contoh dan pesan-pesan yang melewati objek-objek ini di dalam *use case* Martin Fowler (2005:81). Simbol-simbol yang digunakan pada *use case* diagram ditunjukan pada Tabel 2.4.

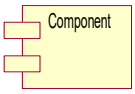
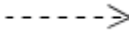
Tabel 2.4 Simbol-Simbol *Sequence Diagram*

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.

2.11.5. *Component Diagram*

Component merupakan bagian fisik dari sebuah sistem, karena menetap di komputer tidak berada di analis. *Component* terhubung melalui antarmuka yang digunakan dan dibutuhkan Martin Fowler (2005:189). *Component* merupakan implementasi *software* dari sebuah atau lebih *class*. *Component* dapat berupa *source code*, komponen biner, atau *executable component*. Sebuah komponen berisi informasi tentang *logic class* atau *class* yang diimplementasikan sehingga membuat pemetaan dari *logical view* ke *component view*. Sehingga *component diagram* merepresentasikan dunia riil yaitu *component software* yang mengandung *component*, *interface* dan *relationship*. Simbol-simbol yang digunakan pada *use case* diagram ditunjukan pada Tabel 2.5.

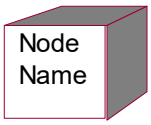
Tabel 2.5. Simbol-simbol *Component Diagram*

No	GAMBAR	NAMA	KETERANGAN
1		*Component*	Sebuah komponen melambangkan sebuah entitas *software* dalam sebuah sistem. Sebuah komponen dinotasikan sebagai sebuah kotak segiempat dengan dua kotak kecil tambahan yang menempel.
2		*Depedency*	Sebuah *Depedency* digunakan untuk menotasikan relasi antara dua komponen. Notasinya adalah tanda panah putus-putus yang diarahkan kepada komponen tempat sebuah komponen itu bergantung.

2.11.6.*Deployment Diagram*

Deployment Diagram menujukkan susunan fisik sebuah sistem, menunjukkan bagian perangkat lunak mana yang berjalan pada perangkat keras mana Martin Fowler (2005:137). *Deployment Diagram* juga menggambarkan tata letak sebuah system secara fisik, menampakkan bagian-bagian *software* yang berjalan pada bagian-bagian *hardware*, menunjukkan hubungan komputer dengan perangkat (*nodes*) satu sama lain dan jenis hubungannya. Simbol-simbol yang digunakan pada *use case* diagram ditunjukan pada Tabel 2.6.

Tabel 2.6.Simbol-simbol *Deployment Diagram*

NO	GAMBAR	NAMA	KETERANGAN
1		*Component*	Pada deployment *diagram*, komponen-komponen yang ada diletakkan didalam *node* untuk memastikan keberadaan posisi mereka.
2		*Node*	*Node* menggambarkan bagian-bagian *hardware* dalam sebuah sistem. Notasi untuk *node* digambarkan sebagai sebuah kubus 3 dimensi.
3		*Association*	Sebuah *association* digambarkan sebagai sebuah garis yang menghubungkan dua *node* yang mengindikasikan jalur komunikasi antara elemen-elemen *hardware*.

2.12. Kerangka Pemikiran

Keterangan *Diagram* Kerangka Pemikiran :

1. Latar Belakang Masalah

   Mendeskripsikan informasi yang tersusun secara sistematis berkenaan dengan fenomena dan masalah problematik untuk diteliti. Latar belakang dimaksudkan untuk menjelaskan alasan mengapa masalah dalam penelitian ingin diteliti, pentingnya permasalahan dan pendekatan yang digunakan untuk menyelesaikan masalah tersebut.

2. Rumusan Masalah

   Rumusan masalah berupa ruang lingkup masalah, pembatasan dimensi dan analisis variabel yang tercakup didalamnya. Rumusan masalah sekaligus menunjukkan fokus pengamatan didalam proses penelitian.

3. Judul Tugas Akhir

   Setelah dirumuskan suatu masalah maka didapat fokus pengamatan, kemudian mencari solusi terbaik untuk memecahkan masalah sehingga solusi tersebut dapat ditarik sebagai sebuah judul.

4. Pengumpulan Data Tertulis dan Tidak Tertulis

   Mengumpulkan semua data yang dibutuhkan, baik *interview* dengan ketua Prodi Ilmu Komunikasi, observasi atau studi literatur di perpustakaan Universitas Sahid Surakarta.

5. Penguasaan Dasar (Desain Aplikasi)

   Melakukan beberapa percobaan membuat aplikasi sederhana dengan tujuan agar dapat lebih memahami bahasa pemrograman PHP dan *database* MySQL sehingga hasilnya menjadi lebih maksimal.

6. Observasi Aplikasi

   Mengamati dan membandingkan beberapa aplikasi yang sudah ada, baik dari karya ilmiah, buku, atau *internet* yang dapat dijadikan referensi untuk membangun aplikasi.

7. Analisis dan Perancangan Sistem

   Menganalisis dan merancang sistem yang akan dibangun seperti apa, bagaimana desainnya, dan apa saja isinya, sehingga sistem ini dapat membantu memecahkan permasalahan yang ada pada *website* Program Studi Ilmu Komunikasi Universitas Sahid Surakarta.

8. Implementasi

   a) Perancangan *Database*

      Membuat *database* dari data-data yang telah didapatkan sesuai dengan kebutuhan *website*.

   b) Perancangan Website Program Studi Ilmu Komunikasi Universitas Sahid Surakarta.

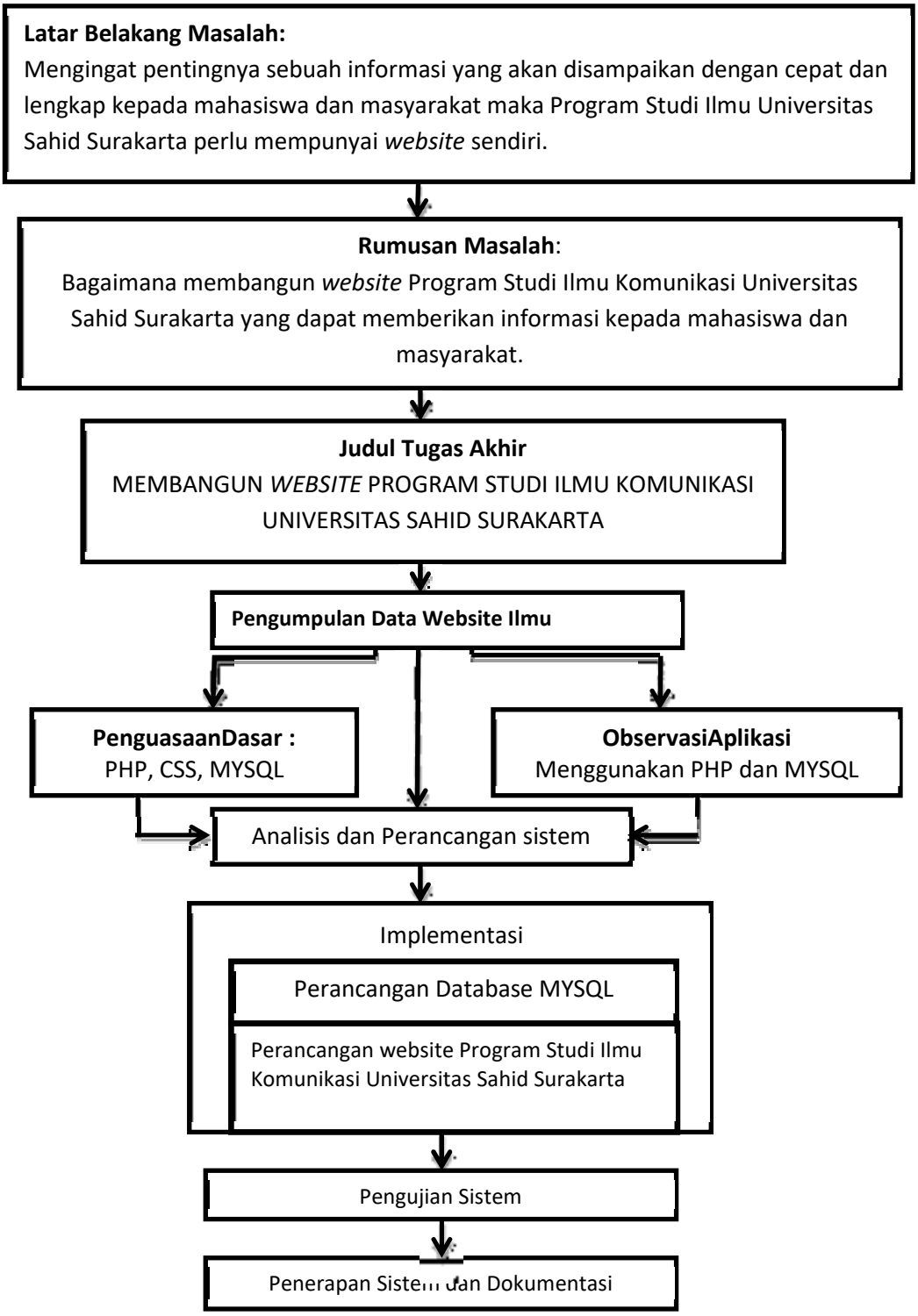
      Membuat *website* dengan dasar *database* yang telah selesai dibuat.

9. Pengujian Sistem

   Pengujian dilakukan di Universitas Sahid Surakarta, tujuannya adalah untuk mengetahui jika ternyata masih ada kesalahan atau kekurangan pada aplikasi.

10. Penerapan Sistem dan Dokumentasi

    Setelah melewati pengujian sistem, maka sistem tersebut dapat diterapkan atau dipakai dan membuat dokumentasi dari keseluruhan kegiatan penyusunan tugas akhir.



Gambar 2.4. kerangka pemikiran

2.13. Jaminan Kualitas Perangkat Lunak

2.13.1. Pengujian *Black Box*

Black box testing adalah metode pengujian perangkat lunak yang tes fungsionalitas dari aplikasi yang bertentangan dengan struktur internal atau kerja. Pengetahuan khusus dari kode aplikasi atau struktur internal dan pengetahuan pemrograman pada umumnya tidak diperlukan.*Black box testing* terfokus pada spesifikasi fungsional dari perangkat lunak (Putu Sudharyana, Dkk, 2012:1).

2.13.2. Klasifikasi *Black box Testing*

Klasifikasi *black box testing* mencakup beberapa pengujian, yaitu:

1. Pengujian Fungsional *(functional testing)*

   Pada jenis pengujian ini, perangkat lunak diuji untuk persyaratan fungsional. Pengujian dilakukan dalam bentuk tertulis untuk memeriksa apakah aplikasi berjalan seperti yang diharapkan. Pengujian fungsional meliputi seberapa baik sistem melaksanakan fungsinya, termasuk perintah-perintah pengguna, manipulasi data, pencarian dan proses bisnis, pengguna layar, dan integrasi. Pengujian fungsional juga meliputi permukaan yang jelas dari fungsi-fungsi, serta operasi back-end (seperti, keamanan dan bagaimana meningkatkan sistem).

2. Pengujian Tegangan *(stress testing)*

   Pengujian tegangan berkaitan dengan kualitas aplikasi di dalam lingkungan. Idenya adalah untuk menciptakan sebuah lingkungan yang lebih menuntut aplikasi, tidak seperti saat aplikasi dijalankan pada beban kerja normal. Pengujian ini adalah hal yang paling sulit, cukup kompleks dilakukan, dan memerlukan upaya bersama dari semua tim.

3. Pengujian Beban *(lood testing)*

   Pada pengujian beban, aplikasi akan diuji dengan beban berat atau masukan, seperti yang terjadi pada situs web, untuk mengetahui apakah aplikasi/situs gagal atau kinerjanya menurun. Pengujian beban beroperasi pada tingkat beban standar, biasanya beban tertinggi akan diberikan ketika sistem dapat menerima dan tetap berfungsi dengan baik.

4. Pengujian Khusus *(ad-hoc testing)*

Jenis pengujian ini dilakukan tanpa penciptaan rencana pengujian *(test plan)* atau kasus pengujian *(test case).* Pengujian khusus membantu dalam menentukan lingkup dan durasi dari berbagai pengujian lainnya dan juga membantu para penguji dalam mempelajari aplikasi sebelum memulai pengujian dengan pengujian lainnya.pengujian ini merukapan metode pengujian formal yang paling sedikit. Salah satu pengguna terbaik dari pengujian khusus adalah untuk penemuan. Pengujian khusus dapat menemukan lubang-lubang dalam pengujian strategi dan dapat mengekspos hubungan diantara subsistem lain yang tidak jelas. Denagn cara ini, pengujian khusus berfungsi sebagai alat untuk memeriksa kelengkapan yang diuji.

5. Pengujian Penyelidikan *(exploratory testing)*

Pengujian penyelidikan mirip dengan pengujian khusus dan dilakukan untuk mempelajari/mencari aplikasi. Pengujian penyelidikan perangkat lunak ini merupakan pendekatan yang menyenangkan untuk penguji.

6. Pengujian Usabilitas *(usalibity testing)*

Pengujian ini disebut juga sebagai pengujian untuk keakraban pengguna *(testing for user-friendliness)*. Pengujian ini dilakukan jika antarmuka pengguna dari aplikasinya penting dan harus spesifik unutk jenis pengguna tertentu. Pengujian usabilitas adalah proses yang bekerja dengan pengguna akhir secara langsung maupun tidak langsung untuk menilai bagaimana pengguna merasakan paket perangkat lunak dan bagaimana mereka berinteraksi. Tujuan dari pengujian usabilitas harus membatasi dan menghilangkan kesulitan bagi pengguna dan untuk memengaruhi area yang kuat untuk usabilitas maksimum.

7. Pengujian Asap *(smoke testing)*

Pengujian ini disebut juga pengujian kenormalan *(sanity testing)* pengujian ini dilakukan untuk memeriksa apakah aplikasi tersebut sudah siap untuk pengujian yang lebih besar dan bekerja dengan baik tanpa cela sampai tingkat yang paling diharapkan.

8. Pengujian Pemulihan *(recovery testing)*

Pengujian pemulihan *(recovery testing)* pada dasarnya dilakukan untuk memeriksa seberapa cepat dan baiknya aplikasi bisa pulih terhadap semua jenis *crash* atau kegagalan hardware, masalah bencana, dan lain-lain. Jenis atau taraf pemulihan ditetapkan dalam persyaratan spesifikasi.

9. Pengujian Volume *(volume testing)*

Pengujian volume adalah pengujian sebuah sistem (baik perangkat keras dan perangkat lunak) untuk serangkaian pengujian dengan volume data yang diproses adalah subjek dari pengujian, seperti sistem yang dapat menangkap sistem pengolahan transaksi penjualan *real-time* atau dapat membarui basis data atau pengembalian data *(data retrieval).*

10. Pengujian Domain *(domain testing)*

Pengujian domain merupakan penjelasan yang paling sering menjelaskan teknik pengujian. Beberapa penulis hanya menulis tentang pengujian domain ketika mereka menulis desain pengujian.

11. Pengujian Skenario *(scenario testing)*

Pengujian skenario adalah pengujian yang realistis, kredibel dan memotivasi stakeholder, tantangan untuk program dan mempermudah penguji untuk melakukan evaluasi. Pengujian ini menyediakan kombinasi variabel-variabel dan fungsi yang sangat berarti daripada kombinasi buatan yang didapatkan dengan pengujian domain atau desain pengujian kombinasi.

12. Pengujian Regresi *(regression testing)*

Pengujian regresi adalah gaya pengujian yang berfokus pada pengujian ulang *(retesting)* setelah ada perubahan. Pada pengujian regresi berorientasi risiko *(risk-oriented regression testing).* Usaha pengujian regresi bertujuan untuk mengurangi risiko berikut ini :

a. Perubahan yang dimaksudkan untuk memperbaiki *bug* yang gagal.
b. Beberapa perubahan memiliki efek samping, tidak memperbaiki *bug* lama atau memperkenalkan *bug* baru.

13. Penerimaan Pengguna *(user acceptance)*

Pada jenis pengujian ini, perangkat lunak akan diserahkan kepada pengguna untuk mengetahui apakah perangkat lunak memenuhi harapan pengguna dan bekerja seperti yang diharapkan. Pada pengembangan perangkat lunak, *user acceptance testing* (UAT), juga disebut pengujian beta *(beta testing),* pengujian aplikasi *(application testing),* dan pengujian pengguna akhir *(end user testing)* adalah tahapan pengembangan perangkat lunak ketika perangkat lunak diuji pada "dunia nyata" yang dimaksudkan oleh pengguna.

14. Pengujian Alfa *(alpha testing)*

Pada jenis pengujian ini, pengguna akan diundang ke pusat pengembangan. Pengguna akan menggunakan aplikasi dan pengembang mencatat setiap masukan atau tindakan yang dilakukan oleh pengguna. Semua jenis perilaku yang tidak normal dari sistem dicatat dan dikoreksi oleh para pengembang.

15. Pengujian Beta *(beta testing)*

Pengujian beta dilakukan setelah pengujian alfa. Versi perangkat lunak yang dikenal dengan sebutan versi beta diliris untuk pengguna yang terbatas di luar perusahaan. Perangkat lunak dilepaskan ke kelompok masyarakat agar dapat memastikan bahwa perangkat lunak tersebut memiliki beberapa kesalahan atau *bug*.
(Jannar Simarmata, 2010:316)