BAB II

LANDASAN TEORI

A. Tracer Study

Tracer study merupakan studi yang tujuan utamanya untuk memperoleh informasi mengenai lulusan yang sudah bekerja dan belum bekerja. Selain itu *tracer study* bertujuan untuk mengetahui hasil pendidikan dalam bentuk penguasaan dan pemerolehan kompetensi lulusan yang diaplikasikan di dunia kerja serta transisi dari dunia pendidikan tinggi ke dunia usaha dan industri. Melalui *tracer study* ini penyelenggara pendidikan dapat mengetahui bagaimana penyelenggaraan dan mutu layanan program melalui penilaian para alumni. Melalui ini, maka penyelenggara mampu untuk memperbaiki dan meningkatkan kualitas layanannya (Nuryake dkk: 2015).

B. Perancangan Sistem Informasi

Perancangan adalah suatu kegiatan yang memiliki tujuan untuk mendesain sistem baru yang dapat menyelesaikan masalah-masalah yang dihadapi perusahaan yang diperoleh dari pemilihan alaternatif sistem terbaik (Karman: 2018). Perancangan sistem mempunyai beberapa tahapan, dimulai dari penentuan metode analisis sistem hingga desain *user interface* sistem.

Terdapat banyak model metode untuk melakukan analisis dan perancangan sistem, namun dalam penelitian ini menggunakan metode *object oriented analysis.* Menurut Mulyani (2016), *object oriented analysis* merupakan teknik analisis sistem dengan menempatkan data dan proses pada sebuah abstraksi yang sering

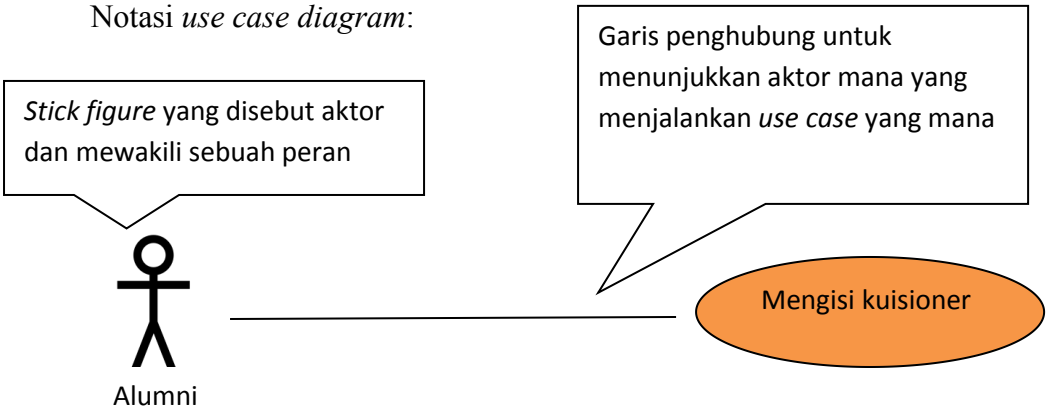
disebut dengan objek dimana setiap objek memiliki atribut dan fungsi yang bisa saling berhubungan untuk menjalankan sistem. O*bject oriented analysis* menggunakan pemodelan *Unified modeling language* (UML).

Unified modeling language (UML) merupakan bahasa pemodelan yang telah terstandarisasi dalam penggunaan metode *object oriented analysis*. Menurut Windu Gata, Grace (2013), *Unified Modeling Language* (UML) adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membanngun perangkat lunak. UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. *Unified modeling language* (UML) menyediakan banyak diagram yang mempunyai fungsi masing dalam menjelaskan sistem informasi, namun dalam penelitian ini tidak semua diagram akan digunakan hanya beberapa diagram saja yang paling banyak digunakan dalam perancangan sistem, diagram tersebut diantaranya sebagai berikut :

1. *Use case diagram*

Use case diagram adalah diagram yang digunakan untuk memodelkan berdasarkan pengguna sistem. U*se case* mendefinisikan aktor atau pengguna yang akan menggunakan sistem dan fitur apa saja yang dapat digunakan oleh aktor tersebut.

Notasi *use case diagram*:



Gambar 2.1 Notasi *Use Case Diagram*

Menurut Triandini dan Suardika (2012) salah satu langkah awal untuk membuat *Use Case Diagram* adalah dengan mengidentifikasi aktor dan proses bisnis dasar. Langkah-langkah membuat *use case diagram*

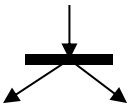
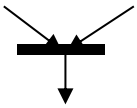
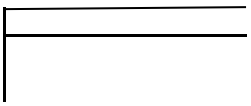
a. Mengidentifikasi aktor. Perhatikan bahwa aktor sebenarnya adalah peran yang dimainkan oleh pengguna. Sebaiknya identifikasi peran spesifik yang dimainkan oleh pengguna. Ingatlah bahwa beberapa orang (pengguna) yang sama mungkin memainkan berbagai peran karena ia menggunakan sistem. Contoh aktor: mahasiswa, dosen, *order clerk*, *department manager*, auditor, dan sebagainya.

b. Setelah peran aktor teridentifikasi, langkah berikutnya adalah menyusun tujuan-tujuan yang ingin dicapai oleh peran-peran tersebut dalam penggunaan sistem. Tujuan tersebut merupakan tugas yang dilakukan oleh aktor untuk mencapai beberapa fungsi bisnis yang memberikan nilai tambah bagi bisnis. Contoh: melihat info biodata, menyimpan data login, mengirim testimoni.

2. *Activity diagram*

Activity diagram digunakan untuk menjelaskan aktivitas atau alur sistem dari awal hingga akhir. Setelah menentukan aktor dan fitur apa saja yang digunakan aktor pada *use case diagram*, pada *activity diagram* ini dijelaskan lebih detail lagi urutan aktivitas ataupun interaksi apa saja yang dilakukan pengguna dengan sistem berdasarkan pembagian fitur aktor masing-masing.

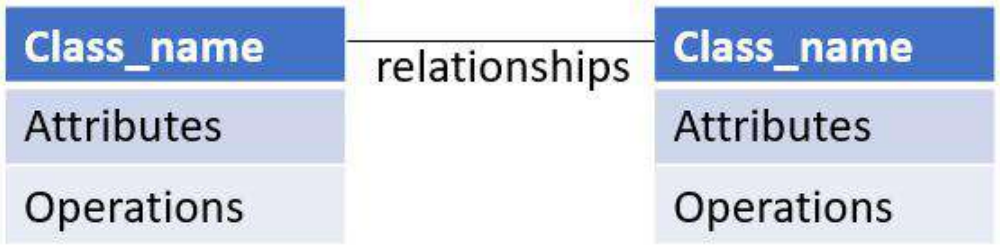
Tabel 2.1 *Activity Diagram*

Gambar	Keterangan
	Start Point, diletakkan pada pojok kiri atas dan merupakan awal aktivitas
	End Point, akhir aktivitas
	Activities, menggambarkan suatu proses/kegiatan bisnis
	*Fork*/percabangan, digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan atau lebih
	*Join* (penggabungan) atau *rake*, digunakan untuk menunjukkan adanya dekomposisi
	Decision Points, menggambarkan pilihan untuk pengambian keputusan, *true* atau *false*
	*Swinlame*, pembagian *activity diagram* untuk menunjukkan siapa melakukan apa

3. *Class diagram*

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah obyek dan merupakan inti dari pengembangan dan desain berorientasi

obyek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metode/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class*, *package*, dan obyek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.



Gambar 2.2 *Class Diagram Elements*

Attributes dan *Operation* (metoda) dapat memiliki salah satu sifat berikut:

a. *Private*, hanya bisa dipanggil dari dalam kelas itu sendiri, *methode/attributes* diawali dengan tanda "-"
b. *Protected,* hanya dapat dipanggil oleh *class* yang bersangkutan dan *class* turunannya, *methode/attributes* diawali dengan tanda "#"
c. *Public*, dapat dipanggil dari semua objek, *methode/attributes* diawali dengan tanda "+"

Tabel 2.2 *Symbol Relationships* antar *Class*

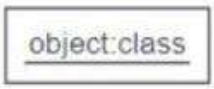
Symbol	**Hubungan**	**Keterangan**
	Asosiasi / *association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*
	Asosiasi berarah/*directed association*	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
	generalisasi	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus)
	Kebergantungan/*dependency*	Relasi antar kelas dengan makna kebergantungan antar kelas
	Agregasi/*aggregation*	Relasi antar kelas dengan makna semua-bagian (*whole-part*)

4. *Sequence diagram*

Sequence diagram adalah *step by step* dari sistem terhadap sebuah *use case*. Dapat disebut sebagai penjabaran sistem dari *use case*. *Sequence diagram*

akan menjelaskan tentang proses yang terjadi dalam sistem. Pembuatan *sequence diagram* dijabarkan berdasarkan *use case* yang ada.

Tabel 2.3 *Symbol Sequence Diagram*

Symbol	**Keterangan**
	Actor, fungsinya sama dengan aktor yang ada di *use case diagram* yakni menandai bahwa ini adalah orang atau sistem luar yang memiliki hubungan dengan objek yang ada di dalamnya. Penggunaannya tidak hanya diperuntukkan untuk orang saja, namun digunakan pula untuk sistem lain. Intinya simbol ini digunakan untuk suatu objek yang terhubung dengan sistem yang akan dibuat nanti.
	Lifeline, digunakan untuk menandakan bahwa adanya tingkah laku pada aktor atau objek sistem yang dirujuk.
	*Object*, untuk menandakan objek-objek yang ada di dalam diagram.
	Lifetime, berbentuk kotak persegi panjang dengan arah ke bawah dan terletak di garis hidup suatu objek atau *class* tanpa menyertakan *text* (tanpa nama).
	Call, digunakan untuk memanggil *class* sesuai dengan *type* keterangan yang ada pada simbol ini.
	Create, digunakan untuk membuat suatu objek lain dalam *sequence*.
	Destroy, digunakan untuk memutus arus atau mematikan garis hidup yang ada pada objek atau *class*.

C. Pengujian Sistem

Pengujian sistem merupakan tahap terakhir dalam pembuatan sistem informasi, pada tahap ini dilakukan pengecekan kesesuian sistem dengan proses bisnis alur sistem yang telah dirancang sebelumnya. Penting sekali dilakukan pengujian sistem, karena dengan dilakukannya pengujian dapat diketahui kesalahan-kesalahan maupun kekurangan pada sistem.

Adapun metode pengujian sistem menurut Binus University dalam situsnya http://scdc.binus.ac.id/himsisfo/2016/10/perbedaan-white-box-testing-dan-black-box-testing/, dijelaskan sebegai berikut :

1. *White Box Testing*

White box testing adalah pengujian yang dilakukan sampai kepada detail pengecekan kode program. Pengujian ini yang didasarkan pada detail prosedur dan alur logika kode program. Pada kegiatan *white box testing*, *tester* melihat *source code* program dan menemukan *bug* dari kode program yang diuji.

Kegiatan *tester* dengan menggunakan metode *white box testing* adalah melihat kode program dan membuat *test case* untuk mencari kesalahan/*bugs*/*error* dari kode program yang dibuat oleh *programmer*.

2. *Black Box Testing*

Pengujian yang didasarkan pada detail aplikasi seperti tampilan aplikasi, fungsi-fungsi yang ada pada aplikasi, dan kesesuaian alur fungsi dengan bisnis

proses yang diinginkan oleh *customer*. Pengujian ini tidak melihat dan menguji *source code* program.

Kegiatan *tester* dengan menggunakan metode *white box testing* adalah membuat *test case* untuk menguji kesesuaian alur kerja suatu fungsi di aplikasi dengan *requirement* yang dibutuhkan *customer* untuk fungsi tersebut dan mencari *bugs*/*error* dari tampilan (*interface*) sistem.

D. Sistem Informasi

Menurut Kadir (2014) sesungguhnya, yang dimaksud dengan sistem informasi tidak harus melibatkan komputer. Sistem informasi yang menggunakan komputer biasa disebut **sistem informasi berbasis komputer** (*Computer-Based Information System* atau CBIS). Dalam praktik, istilah sistem informasi lebih sering dipakai tanpa embel-embel berbasis komputer walaupun dalam kenyataannya komputer merupakan bagian yang penting.

E. Website

Website merupakan kumpulan halaman-halaman yang digunakan untuk menampilkan informasi teks, gambar diam atau gerak, animasi, suara, dan atau gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian bangunan yang saling terkait, yang masing-masing dihubungkan dengan jaringan-jaringan halaman (Bekti: 2015).

Situs atau web dapat dikategorikan menjadi dua yaitu website statis dan website dinamis. *Web* statis adalah *web* yang berisi/menampilkan informasi-

informasi yang sifatnya statis (tetap). Disebut statis karena suatu *web* hanya berhubungan dengan halaman *web* lain dan berisi suatu informasi yang tetap. Sebagai contoh *web* statis adalah website profil perusahaan. Sedangkan web dinamis adalah web yang menampilkan informasi serta dapat berinteraksi dengan pengguna. *Web* yang dinamis memungkinkan pengguna untuk beriteraksi menggunakan form sehingga dapat mengolah informasi yang ditampilkan. *Web* dinamis bersifat interaktif, tidak kaku, dan terlihat lebih indah. Contoh dari website dinamis adalah *website* toko online dimana pemilik *website* dapat melakukan perubah-perubahan produk yang kemudian perubahan tersebut tersimpan pada *database.*

F. Pemrograman *Web*

PHP atau *Hypertext Prepocessor* merupakan bahasa pemrograman berbasis *web* yang memiliki kemampuan untuk memproses data *dinamis*. Artinya, dapat membentuk suatu tampilan berdasarkan pemintaan terkini (Kadir 2008). *Script* PHP harus selalu di awali dengan <? atau <?php dan diakhiri dengan ?> seperti contoh berikut : **<?php** tempat Penulisan script PHP **?>**

Seiring dengan semakin berkembangnya waktu, teknologi *website* juga terus berubah mengiringi perkembangan jaman. Dengan adanya perubahan teknologi akan semakin memudahkan dalam pembuatan *website* dan mempercepat dalam proses pembuatanya. Para *programmer* dimudahkan dengan adanya *framework* yang disediakan secara gratis oleh *vendor-vendor* yang tersedia di situs *internet*.

Framework adalah kerangka kerja. *Framework* juga dapat diartikan sebagai kumpulan *script* (terutama *class* dan *function*) yang dapat membantu *developer/programmer*, seperti koneksi ke *database*, pemanggilan *variable, file,* dan lain-lain sehingga pekerjaan *developer* lebih fokus dan lebih cepat dalam membangun aplikasi.

Dalam penelitian ini penulis menggunakan *framework codeigniter* dalam pembuatan Sistem Informasi *Tracer Study* Alumni dan Penilaian *Stakeholders* Berbasis *Web*, menurut Rahmawati (2017) dalam bukunya yang berjudul *Codeigniter Web Framework* menjelaskan bahwa *codeigniter* merupakan sebuah *framework* PHP dengan konsep MVC *(Model, View, Controller)* yang dapat memudahkan *developer* (pengembang) untuk membuat aplikasi *web* dengan cepat, karena konsep MVC ini memisahkan antara *query* ke *database* (Model) dengan tampilan (*View*) serta logika pemrograman (*Controller*). Kelebihan menggunakan *framework codeigniter* antara lain adalah :

1. Gratis, *framework codeigniter* dapat dengan mudah diunduh dalam *website* resminya https://www.codeigniter.com,
2. Menghemat waktu, karena pengembangan tinggal menggunakan *library* yang telah disediakan *framework codeigniter*,
3. Menggunakan konsep MVC, konsep MVC memisahkan antara *query database* (Model), desain tampilan (*View*) dan logika sistem (*Controller*) sehingga memudahkan dalam pengembangan kedepannya
4. Dokumentasi yang lengkap tersedia saat mengunduh, namun juga dapat dengan mudah dilihat pada *website* resminya,

5. Komunitas, *framework codeigniter* telah banyak digunakan oleh pengembang *web* sehingga memudahkan untuk berinteraksi bertanya dan berdiskusi.

G. Database

Database atau basis data merupakan suatu kumpulan data terhubung yang disimpan secara bersama-sama pada suatu media, yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu, dan dengan software untuk melakukan manipulasi untuk kegunaan tertentu. Basis data bisa diartikan juga sebagai sekumpulan data yang disusun dalam bentuk beberapa table yang saling memiliki relasi maupun berdiri sendiri. (Pamungkas, 2017),

Menurut Huda (2019) MySQL adalah sebuah perangkat lunak sistem manajemen 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 di bawah lisensi GNU *General Public License* (GPL), tetapi mereka juga menjual di bawah lisensi komersial untuk kasus-kasus yang bersifat khusus. Sebagai *database server*, MySQL dapat dikatakan lebih unggul dibandingkan database *server* lainnya, terutama dalam kecepatan. Berikut ini beberapa keistimewaan MySQL antara lain:

1. Portabilitas

MySQL dapat berjalan stabil pada berbagai sistem operasi seperti windows, linux dan lainnya.

2. *Multiuser*

MySQL dapat digunakan oleh beberapa user dalam waktu yang bersamaan tanpa mengalami masalah atau konflik.

3. *Security*

MySQL memiliki beberapa lapisan sekuritas seperti level subnetmask, nama host, dan izin akses user dengan sistem perizinan yang mendetail serta password terenkripsi.

4. *Scalability and limits*

MySQL mampu menangani database dalam skala besar, dengan jumlah record lebih dari 50 juta dan 60 ribu tabel serta 5 milyar baris.